

**PENGARUH TRANSFORMASI RUANG WARNA TERHADAP
AKURASI EKSTRAKSI FITUR DALAM DETEKSI KENDARAAN**

SKRIPSI

Oleh:

MUHAMMAD ZULKIFLI HARDIAN

165090300111021



**JURUSAN FISIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG
2020**

**PENGARUH TRANSFORMASI RUANG WARNA TERHADAP
AKURASI EKSTRAKSI FITUR DALAM DETEKSI KENDARAAN**

SKRIPSI

Sebagai Salah Satu Syarat Untuk Meraih Gelar
Sarjana Sains Bidang Fisika

Oleh:

MUHAMMAD ZULKIFLI HARDIAN

165090300111021



**JURUSAN FISIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG
2020**

LEMBAR PENGESAHAN SKRIPSI

PENGARUH TRANSFORMASI RUANG WARNA TERHADAP AKURASI EKSTRAKSI FITUR DALAM DETEKSI KENDARAAN

Oleh:

MUHAMMAD ZULKIFLI HARDIAN

165090300111021

Setelah dipertahankan di depan Majelis Penguji

Pada tanggal **16 JULI 2020**.....

Dan dinyatakan memenuhi syarat untuk memperoleh gelar
Sarjana Sains dalam bidang Fisika

Pembimbing I

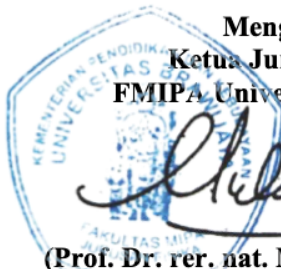
Pembimbing II




(Dr. Eng. Agus Naba, S.Si, MT. Ph.D)
NIP. 197208061995121001



(Dr. rer. nat. Abdurrouf, S.Si., M.Si.)
NIP. 197209031994121001



Mengetahui,
Ketua Jurusan Fisika
FMIPA Universitas Brawijaya



(Prof. Dr. rer. nat. Muhammad Nurhuda)
NIP. 196400910199021001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Muhammad Zulkifli Hardian
NIM : 165090300111021
Jurusan : Fisika
Penulis Skripsi Berjudul :

PENGARUH TRANSFORMASI RUANG WARNA TERHADAP AKURASI EKSTRAKSI FITUR DALAM DETEKSI KENDARAAN

Dengan ini menyatakan bahwa:

1. Isi dari skripsi yang saya tulis dan saya buat adalah benar-benar karya saya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam skripsi ini.
2. Apabila di kemudian hari ternyata skripsi yang saya tulis terbukti hasil menjiplak, maka saya akan bersedia menanggung segala resiko yang akan saya terima.

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 3 Maret 2020

Yang menyatakan,



Muhammad Zulkifli Hardian

165090300111021

ABSTRAK

Klasifikasi gambar adalah aplikasi mendasar dalam *computer vision* yang dapat diimplementasikan untuk deteksi obyek. *Computer vision* banyak digunakan diantaranya pada sistem klasifikasi mobil, penyakit maupun dalam bidang industri. *Convolutional Neural Network (CNN)* biasa digunakan untuk mengekstraksi fitur dari citra input yang diberikan, untuk saat ini penggunaan *deep neural network* telah menunjukkan kinerja terbaik untuk klasifikasi gambar. sebagian dataset terdiri dari sejumlah gambar berwarna dengan satu ruang warna, yaitu RGB tanpa dilakukannya modifikasi. Pada penelitian ini mencoba untuk mengeksplorasi pentingnya penggunaan ruang warna dan menunjukkan bahwa ruang warna (yang berbasis transformasi citra dengan ruang warna RGB) dapat mempengaruhi akurasi ekstraksi fitur dan klasifikasi. Penelitian ini menggunakan 2 kelas berupa data positif dan negatif yang memiliki warna dan pencahayaan cukup untuk proses *training*, kemudian menggunakan data uji berupa warna kendaraan dan pencahayaan sekitar yang berbeda-beda guna mengetahui ruang warna yang dapat bekerja dengan baik. Ruang warna lain mengambil citra RGB sebagai input, lalu dikonversi menjadi 5 ruang warna yang berbeda, lalu citra tersebut digunakan menjadi input. Penelitian ini menggunakan 3 model *CNN* dengan beda jumlah filter maupun layer *CNN* yang digunakan dengan harapan dapat memberikan perbedaan yang signifikan antara transformasi kanal warna yang terjadi.

Kata kunci: *Computer vision*, *CNN*, kanal warna, ruang warna, klasifikasi citra

ABSTRACT

Image classification is a fundamental application in computer vision that can be implemented for object detection. Computer vision is widely used including in the classification system of cars, diseases and in industry. Convolutional Neural Network (CNN) is used to extract features from a given input image. The use of deep neural networks has shown the best performance for image classification. A portion of the dataset consists of a number of color images with one color space, i.e. RGB without modification. This research tries to explore the importance of using color space and show that color space (which is based on image transformation with RGB color space) can affect the accuracy of feature extraction and classification. This study uses 2 classes in the form of positive and negative data that have sufficient color and lighting for the training process, then use test data in the form of different vehicle colors and ambient lighting to determine which color space can work well. Other color spaces take RGB images as input, then convert to 5 different color spaces, then the image is used as input. This study uses 3 CNN models with different number of filters and CNN layers used in the hope that it can provide a significant difference between the color channel transformations that occur.

Keywords: Computer vision, CNN, color channels, color space, image classification

KATA PENGANTAR

Bismillahirrahmanirrahim, Alhamdulillah robil ‘alamin. Segala puji dan syukur kepada Allah Subhanahu wa Ta’ala, Tuhan semesta alam yang telah mencurahkan karunia dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul **“PENGARUH TRANSFORMASI RUANG WARNA TERHADAP AKURASI EKSTRAKSI FITUR DALAM DETEKSI KENDARAAN”** sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Sains dalam bidang Fisika.

Dengan selesainya karya tulis ini, penulis ingin mengucapkan terima kasih kepada pihak-pihak yang memberikan banyak masukan dan dorongan:

1. Hendra Mardian, Almh. Dra. Yuni Trisnarningsih, Seila Munadzorifa dan Rani Mardianingsih selaku kedua orang tua tercinta dan saudara kandung penulis.
2. Dr. Eng. Agus Naba, S.Si, MT., Ph.D., sebagai dosen pembimbing I dan Dr.rer.nat. Abdurrouf, S.Si., M.Si. selaku pembimbing II yang telah memberikan pengarahan dan masukan kepada penulis selama penyusunan skripsi.
3. Udacity, Jason Brownlee, Ph.D., Google, Medium, ScienceDirect, Github, Towardsdatascience karena telah menunjang penulisan skripsi ini.
4. Wulan Tri Dayanti yang sudah menjadi *support-system* selama perkuliahan.
5. Sinatryo Abikusumo, Radite A.Y, Khozi Z.A, Majdi R.A dan Naufal Diva R selaku teman – teman penulis yang senantiasa menjadikan masa perkuliahan yang menyenangkan dan berkesan.
6. Riyan Al-royan, Rakha Perdana, Teguh R, Irfan Maula, M. Iqbal, M Zholla Fanani selaku sahabat saya yang selalu memberi dukungan kepada saya
7. Pembaca yang telah meluangkan waktunya untuk membaca skripsi ini.

Penulis berharap pembuatan karya tulis ini dapat dijadikan sebuah pembelajaran dan pengembangan untuk kedepannya dengan masihnya banyak kekurangan, penulis berharap bahwa pembaca memberikan kritik dan saran yang bersifat membangun

Semoga skripsi ini dapat memberikan manfaat dan inspirasi yang dapat memberikan sumbangan bagi kemajuan ilmu pengetahuan serta peradaban manusia yang baik.

Malang, 3 Maret 2020

Penulis

DAFTAR ISI

LEMBAR PENGESAHAN SKRIPSI	i
LEMBAR PERNYATAAN	iii
ABSTRAK.....	v
ABSTRACT	vii
KATA PENGANTAR.....	ix
DAFTAR ISI	xi
DAFTAR GAMBAR.....	xiii
DAFTAR TABEL	xiv
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan Penelitian.....	2
1.5 Manfaat Penelitian	2
BAB II TINJAUAN PUSTAKA.....	3
2.1 Computer Vision.....	3
2.2 Deteksi Obyek	3
2.3 Citra Digital.....	4
2.4 Elemen – elemen Citra Digital.....	5
2.5 Ruang Warna.....	6
2.5.1 Red Green Blue (RGB)	7
2.5.2 Grayscale.....	8
2.5.3 Hue Saturation Value (HSV).....	9
2.5.4 Cyan Magenta Yellow Black (CMYK)	11
2.5.5 YCbCr.....	12
2.5.6 XYZ.....	13
2.6 Artificial Intelligence	14
2.7 Machine Learning.....	15
2.8 Supervised Learning.....	16
2.9 Artificial Neural Networks.....	17

2.10	Convolutional Neural Networks	18
2.11	Python.....	19
2.12	OpenCV	19
2.13	Tensorflow	20
2.14	Keras	20
BAB III METODE PENELITIAN		21
3.1	Lokasi dan Waktu	21
3.2	Alat dan Bahan	21
3.3	Tahapan Penelitian.....	21
3.3.1	Persiapan Perangkat Lunak.....	21
3.3.2	Pengambilan Data.....	23
3.3.3	Konversi Ruang Warna	24
3.3.4	Pembuatan arsitektur <i>Neural Network</i>	24
3.3.5	Pelatihan Model Neural Network (<i>training</i>)	27
3.3.6	Pengujian model Neural Network (<i>testing</i>).....	28
BAB IV HASIL DAN PEMBAHASAN.....		30
4.1.	Model Convolutional Neural Network.....	30
4.2.	Transformasi Ruang Warna	31
4.3.	Prediksi tiap ruang warna untuk citra dengan intensitas pencahayaan berbeda.....	36
BAB V PENUTUP		41
5.1	KESIMPULAN	41
5.2	SARAN.....	41
DAFTAR PUSTAKA.....		42
LAMPIRAN A DATA HASIL PENELITIAN		45
LAMPIRAN B PROGRAM.....		64

DAFTAR GAMBAR

Gambar 2. 1 Struktur umum deteksi obyek	4
Gambar 2. 2 Spektrum Warna Cahaya Tampak	7
Gambar 2. 3 Wama RGB dalam ruang berdimensi tiga	8
Gambar 2. 4 Palet Grayscale pada 3 kanal warna RGB.....	8
Gambar 2. 5 Citra ruang warna RGB yang telah dikonversi ke ruang warna Grayscale.....	9
<i>Gambar 2. 6 Model Ruang warna HSV</i>	<i>9</i>
Gambar 2. 7 Model Ruang warna CMYK dimana tiap gabungan kanal warna tersebut membuat warna yang lebih gelap	12
<i>Gambar 2. 8 model ruang warna YCbCr</i>	<i>12</i>
Gambar 2. 9 model ruang warna XYZ.....	13
<i>Gambar 2. 10 Kategorisasi Kecerdasan buatan</i>	<i>15</i>
Gambar 2. 11 Perbandingan AI, Machine Learning dan Deep Learning....	15
Gambar 2. 12 Visual object recognition pada deep learning.....	16
Gambar 2. 13 Alur Kerja Supervised Learning	17
Gambar 2. 14 Skema <i>artificial neural networks</i>	18
Gambar 2. 15 Contoh CNN dengan banyak lapisan.....	18
Gambar 3. 1 tampilan home google colab.....	22
Gambar 3. 2 tampilan shell google colab.....	22
Gambar 3. 3 struktur data tiap ruang warna	24
Gambar 3. 4 Arsitektur model Convolution Neural Network-3 Layer.....	25
Gambar 3. 5 Arsitektur model Convolution Neural Network-7 Layer.....	25
Gambar 3. 6 Arsitektur model Convolution Neural Network- 10 Layer	26
Gambar 3. 7 Categorical Crossentropy.....	27
Gambar 4. 1 tampilan array dan konversi citra grayscale.....	32
Gambar 4. 2 tampilan array dan konversi citra HSV	33
Gambar 4. 3 tampilan array dan konversi citra XYZ.....	33
Gambar 4. 4 tampilan array dan konversi citra YCrCb.....	34
Gambar 4. 5 tampilan array dan konversi citra CMYK.....	34

DAFTAR TABEL

Tabel 2. 1 Nilai Hue terhadap warna.....	11
Tabel 3. 1 Tabel Hasil pada Citra uji.....	29
Tabel 3. 2 Confusion matrix untuk evaluasi model.....	29
Tabel 4. 1 Perbandingan nilai akurasi training dengan 15 epoch pada ruang warna RGB, XYZ, HSV dan YCbCr	30
Tabel 4. 2 perbandingan nilai akurasi training dengan 15 epoch pada ruang warna keabuan dan CMYK	31
Tabel 4. 3 Tabel perbandingan percobaan pada data yang telah diacak	35
Tabel 4. 4 contoh gambar uji pada tiap ruang warna dengan kondisi cahaya berbeda	36
Tabel 4. 5 data hasil prediksi dengan citra uji sebanyak 5 kali percobaan..	37
Tabel 4. 6 sampel prediksi citra pada ruang warna berbeda di siang hari...	38
Tabel 4. 7 Sampel prediksi pada kondisi malam hari.....	39

BAB I

PENDAHULUAN

1.1 Latar Belakang

Klasifikasi citra merupakan sebuah pengaplikasian dasar yang digunakan pada sistem *Computer Vision*. *Computer vision* merupakan ilmu yang menggunakan *image processing* untuk membuat keputusan berdasarkan citra input. Dengan kata lain, *computer vision* bertujuan untuk membangun mesin pandai yang dapat “melihat”. Kerangka kerja umum yang biasa dilakukan dalam *computer vision* adalah: proses akuisisi citra, pra pemrosesan, ekstraksi fitur, klasifikasi dan terakhir pengambilan keputusan [1].

Ekstraksi fitur adalah proses untuk mendapatkan ciri khas dari citra input, sehingga dapat dikenali pada proses *training*. Ekstraksi fitur merupakan dasar dalam sistem klasifikasi dalam deteksi obyek dalam *computer vision*. Salah satunya adalah deteksi jenis kendaraan, manusia maupun barang yang banyak digunakan saat ini [2].

Metode *Convolutional Neural Network* (CNN) banyak digunakan untuk sistem *computer vision*, karena memiliki kualitas akurasi dan kecepatan yang baik dalam ekstraksi fitur dan akurasi klasifikasi citra yang baik. Seringkali sistem *computer vision* menggunakan ruang warna RGB dibandingkan ruang warna lainnya seperti CMYK, Grayscale, HSV, XYZ, YCbCr dan lain-lain.

Penelitian sebelumnya [3] meneliti tentang penggunaan ruang warna lain dalam sistem klasifikasi citra dengan citra input sebanyak 10 kelas. Hasil yang didapat adalah bahwa penggunaan ruang warna mempengaruhi kualitas akurasi pada tiap kelas yang berbeda. Namun pada sistem tersebut tidak mempertimbangkan aspek cahaya pada citra tersebut. Oleh karena itu penelitian ini membahas tentang pengaruh penggunaan ruang warna lain terhadap akurasi ekstraksi fitur yang dilakukan oleh CNN, terutama pada kondisi pencahayaan yang berbeda pada citra uji berupa kendaraan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana model CNN yang akurat untuk ekstraksi fitur?
2. Bagaimana pengaruh transformasi ruang warna terhadap akurasi ekstraksi fitur oleh CNN?
3. Bagaimana kemampuan prediksi tiap ruang warna terhadap citra dengan perbedaan intensitas pencahayaan pada citra uji ?

1.3 Batasan Masalah

Ruang lingkup dalam penelitian ini dibatasi oleh beberapa hal sebagai berikut:

1. Penggunaan klasifikasi 2 kelas, yaitu kendaraan dan bukan kendaraan.
2. Penggunaan jenis kendaraan adalah kendaraan roda 4.
3. Penggunaan Metode model *Convolutional Neural Networks*.

1.4 Tujuan Penelitian

Tujuan Penelitian yang dilakukan adalah:

1. Membangun model CNN yang akurat untuk ekstraksi fitur.
2. Menerapkan dan mengetahui pengaruh transformasi ruang warna terhadap akurasi ekstraksi fitur oleh CNN.
3. Mengetahui kemampuan prediksi tiap ruang warna terhadap citra dengan perbedaan intensitas pencahayaan.

1.5 Manfaat Penelitian

Manfaat yang didapatkan pada penelitian ini adalah:

1. Menerapkan penggunaan model CNN dalam deteksi obyek dengan modifikasi transformasi ruang warna.
2. Memberikan informasi terhadap pengaruh transformasi ruang warna untuk akurasi ekstraksi fitur.
3. Memberikan informasi kemampuan prediksi tiap ruang warna untuk citra yang memiliki intensitas cahaya berbeda.

BAB II

TINJAUAN PUSTAKA

2.1 Computer Vision

Visi komputer (*Computer Vision*) adalah teknologi yang banyak dipakai saat ini. Teknologi visi komputer merupakan bidang dari teknologi *Artificial Intelligence*. *Computer Vision* merupakan metode yang digunakan untuk mendapatkan ciri dari citra yang diambil dari dunia nyata agar dapat mengerti informasi yang terkandung dari citra tersebut. Inti dari teknologi ini yaitu meniru kemampuan penglihatan manusia oleh benda elektronik, sehingga dapat memahami dan mengerti informasi dari citra masukkan [4].

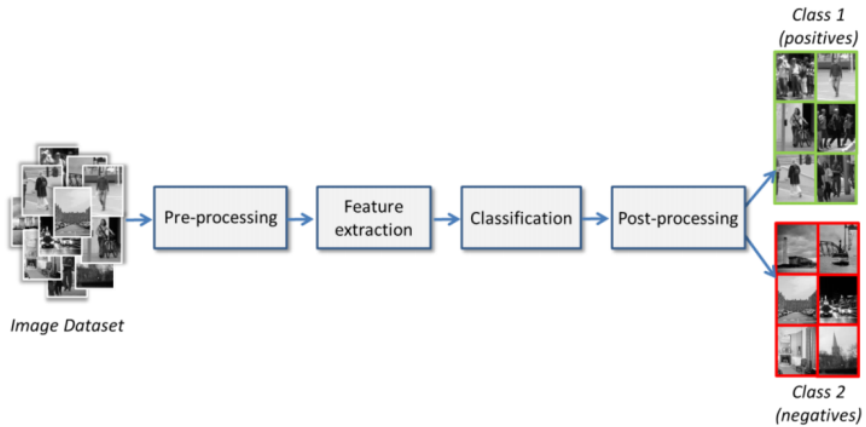
Pemahaman citra oleh komputer dilakukan dengan penguraian informasi dari data citra dengan menggunakan model yang dibangun dengan bantuan geometri, statistika, fisika dan metode lainnya [5].

Computer Vision juga dikenal sebagai teknologi untuk mengotomasi dan mengintegrasikan berbagai proses dan representasi yang menghasilkan persepsi penglihatan. Saat ini *Computer Vision* digunakan untuk mendeteksi wajah pada citra (*Face Detection*), mengenali ekspresi wajah (*facial expression recognition*). Penggunaan *computer vision* banyak digunakan Bersama dengan jaringan saraf tiruan (*artificial neural network*).

2.2 Deteksi Obyek

Deteksi obyek merupakan teknologi komputer untuk mendeteksi dan melokalisir obyek dari kategori yang sudah dikenali. Deteksi obyek banyak digunakan misalnya untuk pengindraan jauh citra geologi bumi, geografi atau lingkungan hidup. Deteksi obyek juga dapat digunakan untuk deteksi gerakan dengan menggunakan CCTV sebagai input dari citra [6].

Sistem deteksi obyek biasanya terdiri dari dua prosedur utama, yaitu ekstraksi fitur dan klasifikasi (lihat gambar 2.1). Langkah ekstraksi fitur yaitu menyandikan tampilan visual citra dalam fitur yang sudah dikenal. Sedangkan langkah klasifikasi yaitu menentukan apakah daerah/citra mengandung *object of interest* atau tidak, dengan menggunakan fitur yang diekstraksi sebelumnya dan *classifier* khusus.



Gambar 2. 1 Struktur umum deteksi obyek

2.3 Citra Digital

Citra digital yaitu citra yang bias diolah komputer, citra digital diwakili dengan matriks yang terdiri dari M kolom N baris. Perpotongan antara kolom dan baris disebut *pixel* (*pixel* = *picture element*), yaitu elemen terkecil dari citra. *Pixel* memiliki 2 parameter, yaitu koordinat dan intensitas atau warna. Nilai pada koordinat (x,y) adalah $f(x,y)$, yaitu besar intensitas atau warna dari *pixel* di titik tersebut. Citra digital dapat ditulis dalam bentuk matriks berikut:

$$\begin{array}{ccccc}
 & F(0,0) & \dots & F(0,M-1) & \\
 F(x,y) & \dots & \dots & F(1,M-1) & (2.1) \\
 & F(N-1,0) & F(N-1,1) & F(N-1,M-1) &
 \end{array}$$

Berdasarkan gambaran tersebut, secara matematis citra digital dapat dituliskan sebagai fungsi intensitas $f(x,y)$, dimana harga x (baris) dan y (kolom) merupakan koordinat posisi dan $f(x,y)$ adalah nilai fungsi pada setiap titik (x,y) yang menyatakan besar intensitas citra atau tingkat keabuan

atau warna dari *pixel* di titik tersebut. Pada proses digitalisasi (sampling dan kuantitas) diperoleh besar baris M dan kolom N hingga citra membentuk matriks $M \times N$ dan jumlah tingkat keabuan *pixel* G [7].

2.4 Elemen – elemen Citra Digital

Citra digital mengandung sejumlah elemen – elemen dasar. Elemen–elemen dasar tersebut dapat dimanipulasi dalam pengolahan citra dan dieksploitasi lebih lanjut dalam *computer vision*. Elemen–elemen tersebut adalah:

- Kecerahan (*brightness*)

Kecerahan merupakan kata lain dari intensitas cahaya, kecerahan pada titik (*pixel*) didalam citra bukanlah intensitas sebenarnya, tetapi intensitas rata-rata dari area yang melingkupinya. Sistem visual manusia mampu menyesuaikan dengan tingkat kecerahan, dari yang paling rendah hingga paling tinggi [8].

- Kontras

Kontras menyatakan sebaran terang (*lightness*) dan gelap (*darkness*) didalam gambar. Citra dengan kontras rendah bercirikan sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Pada citra dengan kontras yang baik, komposisi gelap dan terang tersebar secara merata [8].

- Kontour

Kontour adalah keadaan yang ditimbulkan oleh perubahan intensitas pada pixel–pixel yang bertetangga. Oleh karena itu menghasilkan tepi–tepi sehingga dapat dideteksi (*edge-detection*) [8].

- Warna

Warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh obyek. Setiap warna mempunyai panjang gelombang yang berbeda, dari warna merah yang memiliki panjang gelombang tinggi hingga ungu dengan Panjang gelombang paling rendah [8].

- **Bentuk**

Bentuk merupakan properti intinsik dari obyek tiga dimensi untuk sistem visual manusia, karena hal ini sangat sensitif untuk diperhatikan mata manusia. Pada umumnya citra yang dibentuk mata merupakan 2 dimensi, sedangkan obyek sebenarnya adalah 3 dimensi. Hal ini menjadi tantangan untuk *pre-processing* dan segmentasi untuk merepresentasikannya [8].

- **Tekstur**

Tekstur merupakan distribusi spasial dari derajat keabuan pada sekumpulan pixel–pixel yang bertetangga.

2.5 Ruang Warna

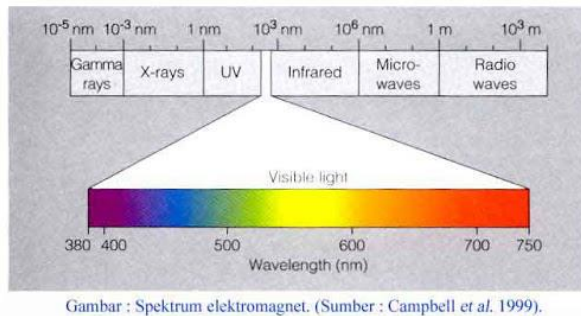
Dalam fisika, radiasi elektromagnetik mengacu pada gelombang atau foton dari medan elektromagnetik yang merambat melalui ruang dengan membawa energi radiasi elektromagnetik. Hal ini termasuk di dalamnya gelombang mikro, inframerah, cahaya tampak, UV, sinar-X dan sinar gamma. Spektrum berbagai gelombang elektromagnetik ditunjukkan pada Gambar 2.2.

Manusia mencoba untuk menentukan koordinat dari tiap ruang warna tersebut agar komputer dapat membantu pekerjaan manusia dengan mempertimbangkan berbagai aspek, sehingga terciptalah ruang warna RGB dan lain-lain untuk merepresentasikan warna yang dilihat manusia.

Warna merupakan karakteristik persepsi visual yang digambarkan oleh kategori warna seperti merah, jingga, kuning, hijau, biru dan ungu. Persepsi warna ini berasal dari stimulasi sel fotoreseptor (khususnya sel kerucut di mata manusia) akibat dari radiasi elektromagnetik (spektrum cahaya tampak). Kategori warna dan spesifikasi fisik warna tergantung melalui panjang gelombang cahaya yang dipantulkan obyek tersebut, refleksi ini diatur oleh sifat fisik obyek seperti penyerapan cahaya, spektrum emisi dan lainnya [9].

Dengan mendefinisikan ruang warna, tiap warna dapat didefinisikan dalam numerik dari koordinatnya, sehingga terjadi beberapa ruang warna seperti RGB, CMYK, XYZ, HSV, YCrCb. Ruang Warna RGB merupakan ruang warna yang mengacu pada sel kerucut yang peka terhadap 3 tipe

panjang gelombang yang merupakan 3 warna dasar (RGB) yaitu Panjang gelombang 564 – 580 nm (merah), 534-545 nm (hijau) dan 420-440 nm (biru) [10].

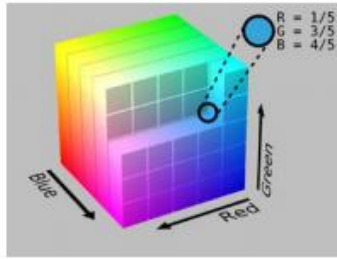


Gambar 2. 2 Spektrum Warna Cahaya Tampak

Citra berwarna merupakan citra yang nilai *pixel*nya merepresentasikan warna tertentu. Banyaknya warna yang mungkin digunakan bergantung kepada kedalaman *pixel* citra yang bersangkutan. Citra berwarna direpresentasikan dalam kanal (*channel*) yang menyatakan komponen komponen warna penyusunnya.

2.5.1 Red Green Blue (RGB)

Setiap *pixel* pada citra warna mewakili warna yang merupakan kombinasi dari 3 warna dasar, yaitu RGB (Red, Green, Blue). Setiap warna dasar menggunakan penyimpanan 8 bit atau 1 byte, yang berarti setiap warna memiliki 255 gradasi warna. Jadi setiap *pixel* mempunyai kombinasi warna sebanyak $2^8 \cdot 2^8 \cdot 2^8 = 2^{24} = 16$ juta warna lebih. Gambar 3.3 menunjukkan ruang warna RGB.



Gambar 2. 3 Warna RGB dalam ruang berdimensi tiga

2.5.2 Grayscale

Citra berskala keabuan (*grayscale*). Citra jenis ini menangani gradasi warna hitam dan putih yang menghasilkan warna abu-abu, dengan memiliki 1 kanal warna yang mendefinisikan warna dari hitam, keabuan dan putih. Dalam hal ini intensitas berkisar antara 0 sampai dengan 255. Nilai 0 menyatakan hitam dan nilai 255 menyatakan putih (lihat Gambar 2.4) [11].



Gambar 2. 4 Palet Grayscale pada 3 kanal warna RGB

Citra *grayscale* memberi kemungkinan warna yang lebih banyak dari pada citra biner (hitam dan putih), karena ada nilai-nilai lain di antara 0 hingga 255, banyaknya kemungkinan minimum dan maksimumnya bergantung pada jumlah bit yang digunakan [12]. Konversi RGB menjadi *grayscale* bias menggunakan rumus berikut :

$$\text{greyscale} = 0.299R + 0.587G + 0.114B \dots\dots\dots (2.2)$$

Konversi nilai tiap kanal warna pada citra tersebut menggunakan perhitungan per *pixel* dimana tiap *pixel* tersebut memiliki nilai pada masing-masing kanalnya (lihat Gambar 2.5).



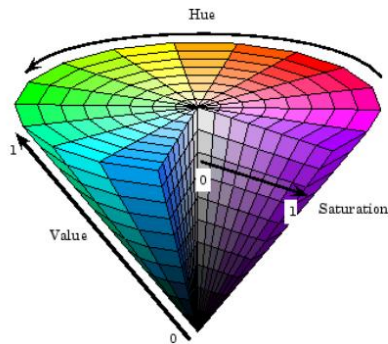
(a)

(b)

Gambar 2. 5 Citra ruang warna RGB yang telah dikonversi ke ruang warna Grayscale

2.5.3 Hue Saturation Value (HSV)

Ruang warna HSV mendefinisikan warna dalam terminologi Hue, Saturation dan Value. Hue menyatakan warna sebenarnya, seperti merah, violet dan kuning. Hue digunakan untuk membedakan warna-warna dan menentukan kemerahan (*redness*), kehijauan (*greenness*), dan sebagainya dari cahaya. Hue berasosiasi dengan panjang gelombang cahaya. Saturation menyatakan tingkat kemurnian suatu warna, yaitu mengindikasikan seberapa banyak warna putih diberikan pada warna. Value adalah atribut yang menyatakan banyaknya cahaya yang diterima oleh mata tanpa memedulikan warna. Gambar 2.6 menunjukkan ruang warna HSV



Gambar 2. 6 Model Ruang warna HSV [13]

Ruang warna HSV menggunakan rumus konversi dengan memanfaatkan nilai kanal warna suatu citra dengan basis RGB, nilai RGB dibagi 255 untuk merubah range nya dari 0 – 255 menjadi 0 – 1.

$$\begin{aligned}
 R' &= R / 255 \quad G' = G/255 \quad B' = B/255 \\
 C_{max} &= \max (R', G', B') \quad C_{min} = \min (R', G', B') \\
 \Delta &= C_{max} - C_{min}
 \end{aligned} \tag{2.3}$$

Perhitungan nilai Hue

$$H = \begin{cases} 0^\circ & \Delta = 0 \\ 60^\circ \times \left(\frac{G' - B'}{\Delta} \text{mod} 6 \right) & , C_{max} = R' \\ 60^\circ \times \left(\frac{B' - R'}{\Delta} + 2 \right) & , C_{max} = G' \\ 60^\circ \times \left(\frac{R' - G'}{\Delta} + 4 \right) & , C_{max} = B' \end{cases} \tag{2.4}$$

Perhitungan nilai Saturasi

$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases} \tag{2.5}$$

Perhitungan nilai Value

$$V = C_{max} \tag{2.6}$$

Nilai Hue adalah derajat dari 0 sampai 360 derajat (lihat Tabel 2.1)

Tabel 2. 1 Nilai *Hue* (warna) dalam derajat ruang warna HSV

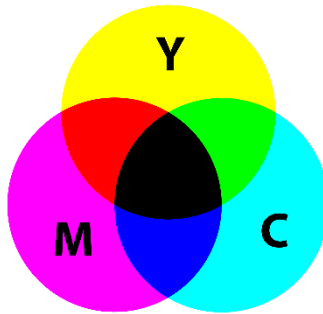
Derajat	Warna
0 - 60	Merah
60 - 120	Kuning
120 - 180	Hijau
180 - 240	Cyan
240 - 300	Biru
300-360	Magenta

Nilai Saturasi mengindikasikan sebagai derajat keabuan dari ruang warna, memiliki rentang dari 0 hingga 100%, bisa juga dengan nilai 0 hingga 1. Jika nilai tersebut 0 maka akan menjadi abu – abu dan apabila 1 ditampilkan warna sesungguhnya. Sehingga dapat disimpulkan jika suatu citra memiliki saturasi rendah, maka derajat keabuan citra tersebut tinggi.

Nilai *Value* merupakan nilai kecerahan dari warna tersebut, memiliki rentang 0 hingga 100% dan bisa juga dengan nilai 0 hingga 1. Jika nilai value 0 maka terlihat hitam, dan apabila nilai value diperbesar akan meningkatkan kecerahan dan menampilkan berbagai warna.

2.5.4 Cyan Magenta Yellow Black (CMYK)

Cyan, *Magenta* dan *Yellow* merupakan warna sekunder dari cahaya dan warna primer dari pigmen. Jika cahaya putih menyinari permukaan yang memiliki pigmen *cyan*, tidak terlihat cahaya merah yang terefleksikan darinya. Tidak seperti ruang warna RGB, ruang warna CMY adalah subtraktif, yang berarti tiap nilai dari kanal warna tersebut berhubungan. Jika tiap kanal di ruang warna CMYK digabung, maka akan menghasilkan warna yang lebih gelap. Warna hitam di ruang warna CMYK merupakan campuran dari 3 kanal warna yang memiliki nilai maksimum (lihat Gambar 2.7).

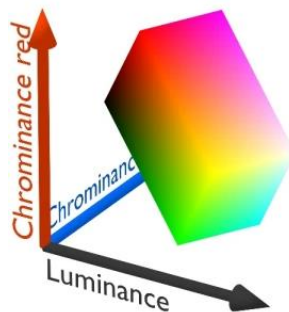


Gambar 2. 7 Model Ruang warna CMYK dimana tiap gabungan kanal warna tersebut membuat warna yang lebih gelap

Model Warna CMYK menggunakan rumus konversi terhadap ruang warna RGB dengan nilai tiap pixel nya yang memuat nilai kanal warna R,G,B dinormalisasi terlebih dahulu nilainya menjadi antara 0 dan 1 (lihat persamaan 2.7).

$$\begin{aligned}
 R' &= (R / 255) & C &= (1-R'-K)/(1-K) \\
 G' &= (G / 255) & M &= (1-G'-K)/(1-K) \\
 B' &= (B / 255) & Y &= (1-B'-K)/(1-K) \\
 & & K &= 1 - \text{MAX}(R',G',B')
 \end{aligned}
 \dots\dots\dots(2.7)$$

2.5.5 YCbCr



Gambar 2. 8 model ruang warna YCbCr

YCbCr merupakan standar internasional bagi pengkodean digital gambar televisi yang didefinisikan di CCIR Recommendation 601 [14].

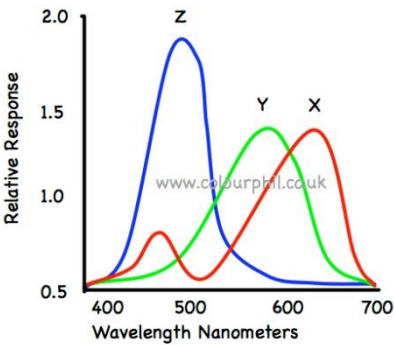
Y merupakan komponen luminance, Cb dan Cr adalah komponen chrominance (lihat Gambar 2.8). Pada monitor monokrom nilai luminance digunakan untuk merepresentasikan warna RGB, secara psikologis mewakili intensitas warna RGB yang diterima oleh mata. Chrominance merepresentasikan corak warna dan saturasi (*saturation*). Nilai komponen ini juga mengindikasikan banyaknya komponen warna biru dan merah pada warna [15].

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,144 \\ -0,169 & -0,331 & 0,500 \\ 0,500 & -0,419 & -0,081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

..... (2.8)

Transformasi RGB ke YCbCr dapat dilihat pada persamaan 2.8

2.5.6 XYZ



Gambar 2. 9 model ruang warna XYZ

Ruang warna CIE XYZ didasarkan pada 'Pengamat Standar' dan 'Penerangan Standar'. Ini adalah model numerik sensitivitas warna

berdasarkan penelitian, dimulai pada 1920, pada sampel orang dengan penglihatan warna normal.

Retina peka terhadap cahaya di bagian belakang mata, yang memiliki tiga jenis reseptor di dekat pusat mata yang dikenal sebagai kerucut. Mereka sensitif terhadap tiga warna dasar, 'merah, hijau dan biru'. Nilai-nilai tristimulus CIE XYZ (lihat Gambar 2.9) diberikan masing-masing ke kurva 'merah, hijau dan biru', mendekati kerucut di mata. Respons relatif masing-masing diplot pada diagram terhadap panjang gelombang dalam nanometer. Mata memiliki sel batang yang posisinya jauh dari pusat retina. Sel batang sensitif terhadap cahaya dengan panjang gelombang rendah dan hanya beroperasi pada tingkat pencahayaan yang rendah.

$$\begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} = \begin{bmatrix} 0.490 & 0.310 & 0.200 \\ 0.177 & 0.813 & 0.011 \\ 0.000 & 0.010 & 0.99 \end{bmatrix} \begin{bmatrix} R_i \\ G_i \\ B_i \end{bmatrix} \dots\dots (2.9)$$

Transformasi RGB ke XYZ dapat dilihat pada Persamaan 2.9

2.6 Artificial Intelligence

Artificial Intelligence atau kecerdasan buatan dibagi ke dalam beberapa aspek dimensi yang direpresentasikan sebagai 4 aspek. Aspek pertama berkaitan dengan bagaimana proses berpikir dan memberikan pendapat. Aspek kedua berkaitan dengan tingkah perilaku. Aspek ketiga berkaitan tentang hal humanisme / manusiawi. Aspek keempat berkaitan tentang hal rasionalitas. Berikut representasi dari *Artificial Intelligence* yang terbagi menjadi 4 aspek penting (lihat Gambar 2.10):

	<i>Humanisme</i>	<i>Rasionalitas</i>
<i>Proses Berpikir</i>	Think like human The exciting new effort to make computers think ... machines with minds , in the full and literal sense. [Haugeland 85].	Think Rationally The study of the computations that make it possible to perceive, reason, and act . [Winston, 1992]
<i>Tingkah Perilaku</i>	Act humanly The study of how to make computers do things at which, at the moment, people are better. [Rich & Knight, 1991]	Act rationally The branch of computer science that is concerned with the automation of intelligent behavior . [Luger and Stubblefield, 1993]

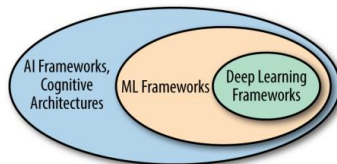
Gambar 2. 10 Kategorisasi Kecerdasan buatan [16]

2.7 Machine Learning

Machine Learning merupakan studi tentang algoritma untuk mempelajari sesuatu dalam melakukan beberapa hal tertentu yang dilakukan oleh manusia secara otomatis. Belajar dalam hal ini berkaitan dengan bagaimana menuntaskan berbagai tugas yang ada, atau membuat suatu prediksi kesimpulan baru yang akurat dari berbagai pola yang sudah dipelajari sebelumnya [17].

Machine Learning adalah salah satu dari cabang AI, dan lebih khusus cabang dari ilmu komputer (*computer science*) yang berhubungan dengan mempelajari sistem dan algoritma yang bisa dipelajari dari data, dan mengganti pengetahuan baru darinya.

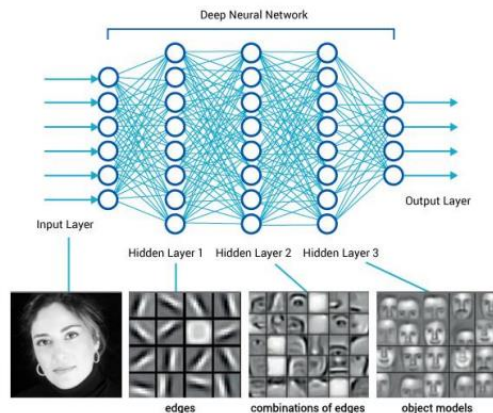
Machine Learning bekerja dari kumpulan data besar dengan mesin yang belajar dari data, mirip dengan cara manusia dan otak biologis memproses informasi.



Gambar 2. 11 Perbandingan AI, *Machine Learning* dan *Deep Learning*

Melalui kerangka kerja *machine learning* didapatkan *deep learning* yang juga diketahui sebagai *Artificial Neural Networks* (ANN) (lihat Gambar 2.11), karena algoritma tersebut dimodelkan pada bagaimana otak memproses data, meskipun saat ini dalam kerangka kerja yang disederhanakan [18].

Deep learning memungkinkan model komputasi yang terdiri dari beberapa *processing layer* untuk mempelajari representasi data dengan berbagai tingkat abstraksi (lihat Gambar 2.12). Metode ini sangat baik dalam pengenalan suara (*speech recognition*), pengenalan obyek visual (*visual object recognition*), deteksi obyek (*object detection*) dan lainnya. *Deep learning* menemukan struktur yang rumit dalam kumpulan data yang besar dengan menggunakan algoritma *backpropagation* untuk menunjukkan bagaimana mesin harus mengubah parameter internalnya yang digunakan untuk menghitung representasi pada setiap lapisan dari representasi pada lapisan sebelumnya [19].



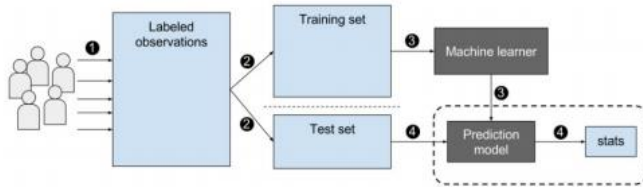
Gambar 2. 12 Visual object recognition pada deep learning

2.8 Supervised Learning

Supervised Learning yang merupakan teknik pembelajaran mesin yang membuat suatu fungsi berdasarkan data latihan yang sudah ada. Dalam

16

hal ini dapat dikatakan untuk teknik ini sudah tersedia data latihan secara detail dan terklasifikasi dengan baik yang dijadikan model data saat dilakukan proses uji coba dengan data tes yang baru dan menghasilkan hasil keluaran yang sesuai dengan diharapkan sebelumnya berdasarkan data latihan yang ada.



Gambar 2. 13 Alur Kerja Supervised Learning

Supervised Learning secara sederhana adalah algoritma pembelajaran yang memiliki data yang telah di labeli dan di training sehingga menjadi model yang dapat diuji kelayakannya dalam pengklasifikasian data baru (lihat Gambar 2.13).

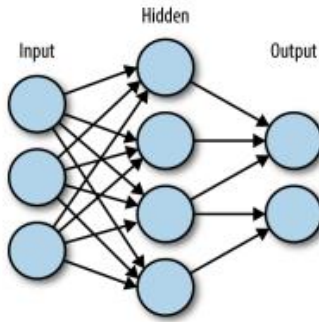
2.9 Artificial Neural Networks

Artificial Neural Networks merupakan upaya manusia untuk memodelkan suatu cara kerja atau fungsi sistem syaraf manusia dalam melakukan tugas tertentu, pemodelan ini didasari oleh kemampuan otak manusia untuk mengorganisir sel – sel neuron.

Artificial Neural Network (ANN) adalah jaringan yang dirancang untuk menyerupai otak manusia yang bertujuan untuk melaksanakan suatu tugas tertentu. Jaringan ini biasanya diimplementasikan dengan menggunakan komponen elektronik atau disimulasikan pada aplikasi komputer [20].

Artificial Neural Network (ANN) adalah sistem prosesor yang terdistribusi secara paralel yang sangat besar dengan memiliki kecenderungan untuk menyimpan pengetahuan yang bersifat pengalaman dan membuatnya siap digunakan. ANN menyerupai otak manusia dalam dua

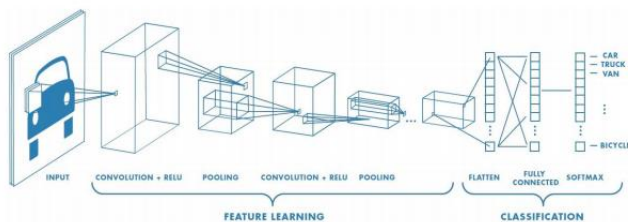
hal yaitu: Pengetahuan yang diperoleh dari sistem jaringan syaraf melalui proses belajar; kekuatan hubungan antar sel syaraf (neuron) yang disebut sebagai bobot sinaptik yang berguna dalam penyimpanan pengetahuan [21].



Gambar 2. 14 Skema *artificial neural networks*

2.10 Convolutional Neural Networks

Convolutional Neural Network (CNN/ConvNet) adalah salah satu algoritma dari *deep learning* yang merupakan pengembangan dari *Multilayer Perceptron* (MLP) yang dirancang untuk mengolah data dalam bentuk dua dimensi, misalnya gambar atau suara. CNN mengklasifikasi data yang terlabel dengan menggunakan metode *supervised learning*. Cara kerja dari *supervised learning* adalah terdapat data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari metode ini adalah mengelompokkan suatu data ke data yang sudah ada [19].



Gambar 2. 15 Contoh CNN dengan banyak lapisan

Secara umum tipe lapisan pada CNN dibagi menjadi dua (lihat Gambar 2.15) yaitu:

- a) Layer ekstraksi fitur (*feature extraction layer*) gambar, letaknya berada pada awal arsitektur tersusun atas beberapa lapisan dan setiap lapisan tersusun atas neuron yang terkoneksi pada daerah lokal (*local region*) dari lapisan sebelumnya. Lapisan jenis pertama adalah *convolutional layer* dan lapisan kedua adalah *pooling layer*. Setiap lapisan diberlakukan fungsi aktivasi dengan posisinya yang berselang-seling antara jenis pertama dengan jenis kedua. Lapisan ini menerima input gambar secara langsung dan memprosesnya hingga menghasilkan output berupa vektor untuk diolah pada lapisan berikutnya.
- b) Layer klasifikasi (*classification layer*) tersusun atas beberapa lapisan, dan setiap lapisan tersusun atas neuron yang terkoneksi secara penuh (*fully connected*) dengan lapisan lainnya. Layer ini menerima input dari hasil keluaran layer ekstraksi fitur gambar berupa vektor kemudian ditransformasikan seperti Multi Neural Networks dengan tambahan beberapa hidden layer. Hasil keluaran berupa akurasi kelas untuk klasifikasi.

2.11 Python

Python dapat digunakan untuk lebih dari sekedar keperluan pemrograman secara umum. Python adalah bahasa yang memiliki potensi luar biasa untuk digunakan dalam domain komputasi ilmiah. Python adalah alat yang efektif untuk digunakan saat menggabungkan komputasi ilmiah dan matematika [22].

2.12 OpenCV

OpenCV (*Open Source Computer Vision Library*) adalah pustaka perangkat lunak yang ditujukan untuk pengolahan citra dinamis secara *real-time*, yang dibuat oleh Intel, dan sekarang didukung oleh Willow Garage dan Itseez. Program ini bebas akses dan berada dalam naungan sumber terbuka

dari lisensi BSD. Pustaka ini merupakan pustaka lintas platform. Program ini didedikasikan sebagian besar untuk pengolahan citra secara real-time.

2.13 Tensorflow

Sistem TensorFlow digunakan untuk machine learning yang sebagian besar karena fleksibilitasnya dan keumuman dalam pemrogramannya. Pendekatan ini mendukung berbagai aplikasi machine learning, termasuk pelatihan dan penyimpulan dengan deep neural network pada sistem yang terdistribusi secara heterogen.

TensorFlow mewakili kerja komputasi yang berdasarkan aliran data graf. Secara khusus, komputasi bisa jadi melakukan 1 atau lebih langkah dalam proses pelatihan (*training*) untuk membuat model *machine learning* atau mungkin penerapan model yang sudah dilatih. Dengan demikian, aliran data graf mendukung proses pelatihan dan penyimpulan.

Aliran data graf terdiri dari node dan edge, dimana setiap node merepresentasikan suatu proses dari operasi, dan nilai mengalir di sepanjang edge. Operasi diimplementasikan oleh kernel yang dapat dijalankan pada jenis perangkat tertentu (misalnya, CPU atau GPU) [23].

2.14 Keras

Keras adalah library dari bahasa pemrograman Python untuk keperluan Deep Learning yang dapat berjalan di atas Theano atau TensorFlow. Keras dikembangkan untuk membuat menerapkan model Deep Learning secepat dan semudah mungkin untuk penelitian dan pengembangan. Keras berjalan pada Python 2 atau 3 dan dapat dijalankan secara mulus pada GPU dan CPU yang diberikan kerangka kerja yang mendasarinya [24].

BAB III

METODE PENELITIAN

3.1 Lokasi dan Waktu

Penelitian dilaksanakan di dua tempat yang berbeda yaitu di Laboratorium Analisa dan Numerik Jurusan Fisika Univ. Brawijaya dan kediaman penulis yang berada di Bunga Kopi Lowokwaru Malang.

3.2 Alat dan Bahan

Alat yang digunakan dalam penelitian adalah Laptop ASUS ROG GL503VD Notebook PC, dengan Spesifikasi Processor Intel(R) Core(TM) i7-7700HQ CPU @ 2.80 GHz, 2400 MHz, 4 Core(s), 8 Logical Processor(s), RAM 24.00 GB, Hard Drive 1000 GB, SSD 128 GB, Kamera Handphone Samsung S9.

Bahan yang digunakan dalam penelitian adalah 1504 citra latih, 370 citra validasi dan 141 citra uji.

3.3 Tahapan Penelitian

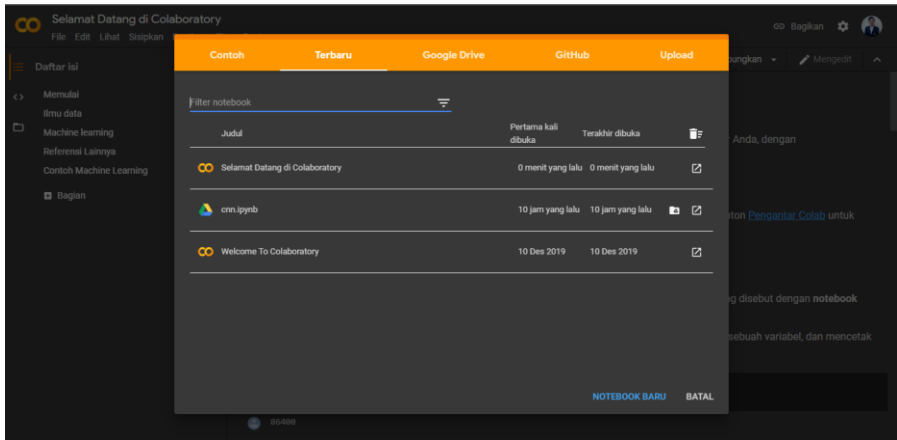
Penelitian ini dilakukan dalam beberapa tahap sebagai berikut:

1. Persiapan Perangkat Lunak (*Software*)
2. Pengambilan Data Citra
3. Konversi ruang warna
4. Pembuatan Arsitektur *Neural Network*
5. Pelatihan *Neural Network* (Training)
6. Pengujian *Neural Network* (Testing)

3.3.1 Persiapan Perangkat Lunak

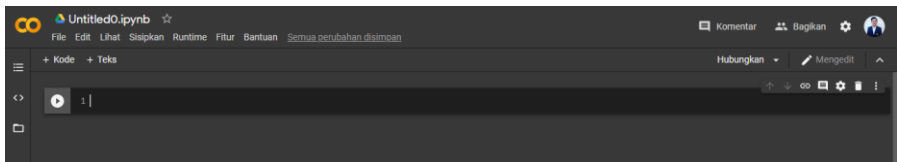
Sebelum memulai penelitian, dilakukan dengan membuat *repository* dengan *cloud computing* yang disediakan *google* yaitu *google colab*, dengan penggunaan *google colab* dapat menghemat *computational cost* sehingga dapat lebih mudah digunakan, terutama karena disediakan GPU gratis sehingga dapat mempercepat proses *training*.

Google colab dapat diakses melalui laman web berikut: <https://colab.google.com/> , setiap script yang dikerjakan dapat langsung dijalankan dengan baik dan dapat dipilih untung *computing engine* berupa CPU, GPU atau TPU (lihat Gambar 3.1).



Gambar 3. 1 tampilan home google colab

Pada tampilan *notebook* berupa *shell* yang dapat dijalankan langsung, pada *shell* tersebut juga dilakukan instalasi modul yang digunakan pada program tersebut (lihat Gambar 3.2).



Gambar 3. 2 tampilan shell google colab

Berikut beberapa library python yang diperlukan:

- Cv2
- Matplotlib

- numpy
- keras

Untuk melakukan instalasi library python gunakan perintah “pip” melalui command prompt sebagai berikut:

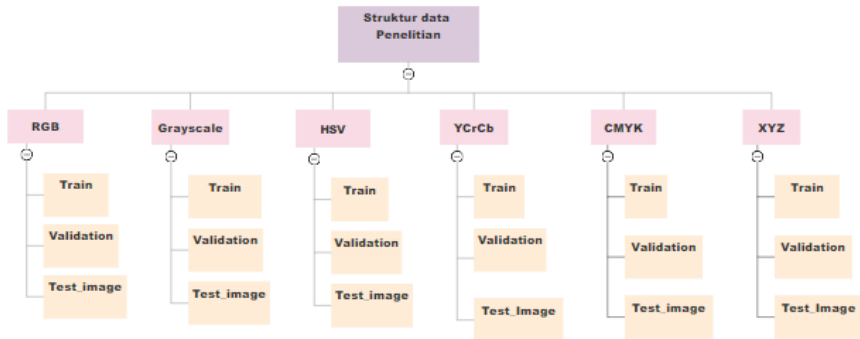
```
pip install opencv-python
pip install keras
pip install matplotlib
pip install numpy
```

setelah semua library terinstall, maka program telah siap untuk dijalankan menggunakan google colab.

3.3.2 Pengambilan Data

Dataset yang didapatkan menggunakan citra yang didapatkan pada mesin pencarian google, citra yang diambil adalah citra positif yaitu citra mobil dan citra negatif yaitu citra pesawat, kucing dan bunga. Hal ini dilakukan dengan mengambil citra sebanyak 1874 citra dengan 2 kelas, dengan struktur data sebanyak 20% data tersebut digunakan untuk validasi data.

Setiap data citra tersebut memiliki ruang warna dasar RGB, citra tersebut dikonversikan ke 5 ruang warna berbeda yaitu grayscale, XYZ, YCrCb, CMYK dan HSV (lihat Gambar 3.3). Tiap ruang warna juga memiliki struktur data yang sama dengan jumlah 1874 citra dengan 2 kelas dan 20% dari data tersebut digunakan untuk validasi data.



Gambar 3. 3 struktur data tiap ruang warna

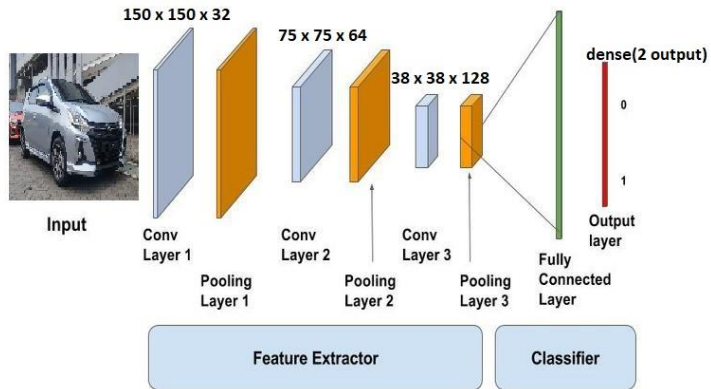
3.3.3 Konversi Ruang Warna

Data citra yang telah dikumpulkan dikonversikan pada tiap ruang warna menggunakan modul openCV, dengan konversi ruang warna tersebut terjadi transformasi nilai pada tiap kanal warna nya, sehingga tiap ruang warna memiliki nilai nya masing – masing yang mempengaruhi proses ekstraksi fitur yang menggunakan CNN.

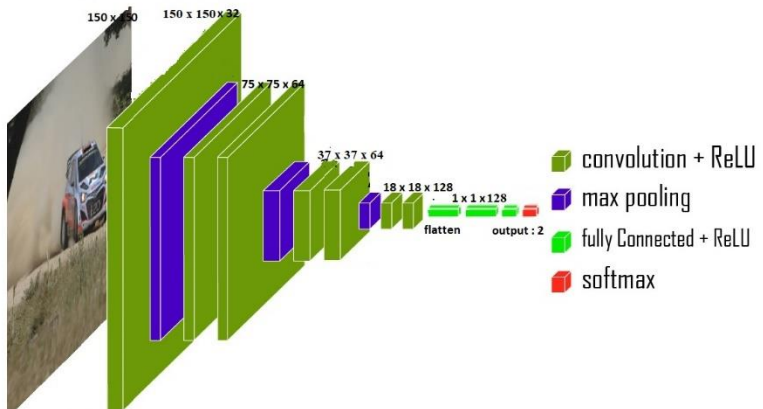
Tiap ruang warna memiliki ciri khas nya masing – masing, seperti ruang warna *grayscale* (keabuan) hanya memiliki 1 kanal warna, sedangkan pada ruang warna RGB, YCbCr, HSV dan XYZ memiliki 3 kanal warna, selain itu ruang warna CMYK memiliki 4 kanal warna, oleh karena itu penentuan penggunaan ruang warna perlu diperhatikan karena tiap ruang warna memiliki ciri masing – masing.

3.3.4 Pembuatan arsitektur *Neural Network*

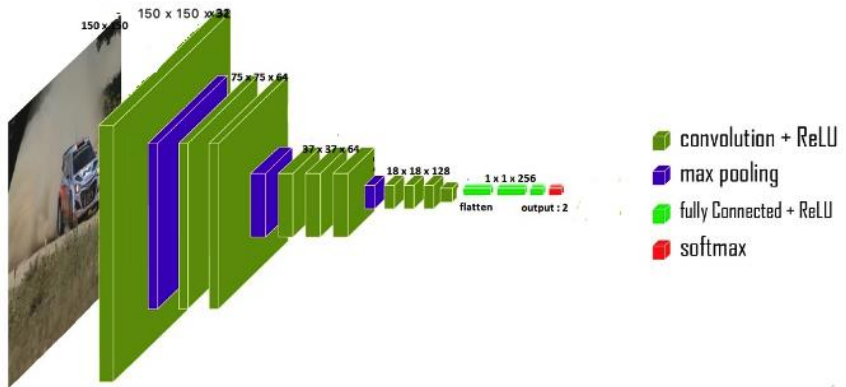
Model *Neural Network* yang digunakan pada penelitian yaitu 3 buah model *Convolutional Neural Network* dengan penggunaan jumlah layer konvolusi yang berbeda - beda, dengan arsitektur sebagai berikut:



Gambar 3. 4 Arsitektur model Convolution Neural Network-3 Layer



Gambar 3. 5 Arsitektur model Convolution Neural Network-7 Layer



Gambar 3. 6 Arsitektur model Convolution Neural Network- 10 Layer

Dengan *image shape* input yang dirubah pada ukuran 150 x 150 pixel, untuk mempermudah proses konvolusi agar memangkas waktu untuk *training*. Digunakan beberapa layer filter konvolusi untuk mengekstraksi fitur dari citra input dan *maxpooling* untuk memperkecil ukuran matriks citra dengan nilai maksimum dari matriks tersebut. Lalu langkah terakhir adalah layer klasifikasi yaitu proses *flatten*, *fully connected* dan *softmax*. Layer klasifikasi berguna untuk mengklasifikasikan tiap neuron yang telah diekstraksi fitur oleh layer – layer sebelumnya.

Penggunaan banyaknya layer konvolusi berbanding lurus dengan ukuran citra input dan banyaknya citra input, karena semakin besar ukuran citra dan banyaknya citra input tersebut maka dibutuhkan layer konvolusi yang lebih banyak, penelitian ini menggunakan beberapa layer CNN dengan 3 kali proses *maxpooling* untuk mendapatkan fitur yang lebih spesifik dari suatu citra.

Penggunaan proses *flatten* digunakan untuk membentuk ulang fitur tersebut menjadi vektor (1 dimensi) agar bisa menjadi input pada *fully connected layer*, dimana pada layer tersebut menghitung skor kelas yang diprediksi, kemudian dilanjutkan pada layer *softmax* yang berguna untuk menghitung probabilitas dari setiap kelas target atas semua kelas target yang memungkinkan, dengan rentang probabilitas antara 0 dan 1.

3.3.5 Pelatihan Model Neural Network (*training*)

Data input yang berjumlah total 1874 dengan 2 kelas di split menjadi 80% untuk data training dan 20 % untuk data validasi, *training set* digunakan dalam pembuatan model machine learning, sedangkan *validation set* untuk menguji performa dan kebenaran dalam model yang dibuat.

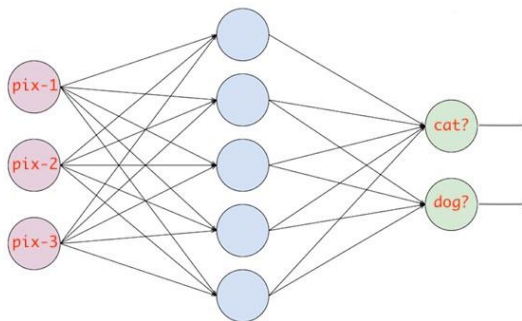
Kernel yang berfungsi sebagai filter yang digunakan untuk proses identifikasi fitur, namun untuk menginisialisasi untuk spesifik bobot terdapat proses yang disebut sebagai *back-propagation* yang didalamnya dibagi menjadi 4 bagian yaitu:

- *Forward Pass*

Pada epoch pertama atau iterasi pelatihan kernel awal dari konvolusi pertama. layer diinisialisasi dengan nilai acak. Jadi setelah output iterasi pertama menjadi sesuatu seperti [1.1.1.1.1.1.1.1.1.1], yang tidak memberikan preferensi ke kelas mana pun karena kernel tidak memiliki bobot khusus.

- *Loss Function – Categorical Crossentropy*

Fungsi yang digunakan pada penelitian ini adalah *categorical crossentropy*, jika neuron tersebut memiliki nilai probabilitas mobil yang lebih tinggi, maka output yang keluar adalah mobil. Secara umum neuron yang memiliki probabilitas kelas yang paling tinggi dari kelas tersebut, maka citra tersebut diklasifikasikan pada kelas tersebut (lihat gambar 3.7).



Gambar 3. 7 Categorical Crossentropy

- *Backward Pass*

Proses penentuan bobot yang paling berkontribusi pada *Loss* dan menemukan cara untuk menyesuaikannya sehingga *Loss* dapat berkurang. dihitung menggunakan dL/dW , di mana *L* adalah kerugian dan *W* adalah bobot dari kernel yang sesuai.

- *Weight Update*

$$w = w_i - \eta \frac{dL}{dW}$$

w = Weight
 w_i = Initial Weight
 η = Learning Rate

..... (3.2)

Merupakan bagian pada *update* bobot pada kernel dengan mengikuti persamaan 3.2

Learning Rate merupakan hal yang perlu diperhatikan dalam *Machine Learning* karena nilai yang lebih besar dari *Learning Rate* mengindikasikan lebih besar langkah untuk optimalisasi dan waktu yang lebih besar untuk konvolusi bobot yang dioptimalkan. Untuk mencapai bobot yang diinginkan, peneliti menggunakan algoritma optimasi, yaitu *Adam Optimizer*. Proses ini dilakukan sebanyak iterasi yang di inginkan.

3.3.6 Pengujian model Neural Network (testing)

Pengujian dilakukan dengan 141 data citra yang sudah diketahui kelas nya dan juga memiliki karakteristik yang berbeda, dengan hasil pelatihan dengan model arsitektur yang sudah ditentukan. Hasilnya disandingkan bersamaan dengan 6 ruang warna yang berbeda dengan nilai akurasi dan juga hasil uji kelas tiap citra tersebut.

Tabel 3. 1 Tabel Hasil pada Citra uji

No.	Name	True Classes	Result					
			RGB	Grayscale	HSV	XYZ	YCrCb	YCrCb
1	airplane_0707	NO						
2	airplane_0708	NO						
...						
...						
...	cat_0865	NO						
...	cat_0866	NO						
...						
...						
...	footage_1	YES						
...	footage_2	YES						
...						
...						
...	color_car	YES						
...						
117	color_car	YES						

Setelah didapatkan tabel hasil citra uji seperti pada tabel 3.1 maka selanjutnya digunakan tabel 3.2 yang berisikan *confusion matrix* untuk evaluasi model yang telah di *training*.

Tabel 3. 2 *Confusion matrix* untuk evaluasi model

	Ruang Warna			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	TP	FN	Total Negatif	60
Negatif	FP	TN	Total	141
Accuracy	(FP+TN)/total		Specificity	TN/(FN+total negatif)
error rate	(FP+FN)/total		Precision	TP/(TP+FP)
Sensitivity	TP/total positif		False Positive	TP/Total Negatif

BAB IV

HASIL DAN PEMBAHASAN

4.1. Model Convolutional Neural Network

Ekstraksi fitur yang dilakukan oleh filter konvolusi dapat terpengaruh oleh adanya transformasi kanal warna maupun perubahan jumlah kanal warna citra tersebut. Terbukti dengan diujinya 3 model arsitektur terjadi perbedaan akurasi proses pelatihan model tersebut, dan hasil grafik tiap ruang warna terdapat pada Lampiran A.

Pada ruang warna yang memiliki 3 kanal warna, perbedaan jumlah layer konvolusi memengaruhi seberapa besar akurasi model pada proses pelatihan. Hal itu terjadi pada ruang warna RGB, XYZ, HSV dan YCbCr. Perbedaan yang terjadi pada proses ekstraksi fitur karena semakin banyaknya filter konvolusi menghasilkan semakin banyak *feature maps* dari suatu citra yang menjadi ciri khas dari suatu kelasnya. Dalam kasus *deep learning*, semakin dalam *Neural Network* maka semakin mendistorsi ruang / citra tersebut. Dapat dilihat pada tabel 4.1 yang berisikan nilai akurasi terhadap proses pelatihan yang telah dibandingkan dengan 3 model CNN dengan perbedaan dalamnya *neural network*.

Tabel 4. 1 Perbandingan nilai akurasi training dengan 15 epoch pada ruang warna RGB, XYZ, HSV dan YCbCr

		Model yang digunakan		
		3 Layer	7 Layer	10 Layer
Ruang Warna	RGB	61,70%	83,68%	74,68%
	XYZ	62,41%	73,76%	50,74%
	HSV	70,92%	80,14%	75,36%
	YCbCr	57,44%	59%	57,44%

Perbedaan yang terjadi dapat diperbaiki dengan banyaknya *epoch* yang diberikan pada proses *training*, lebih banyak layer yang digunakan maka semakin banyak *epoch* dan juga dataset yang digunakan untuk menunjang

model arsitektur yang dirancang. Oleh karena itu, pada proses *pre-processing* perlu diperhatikan bagaimana citra input agar proses *training* dapat berjalan dengan baik.

Pada ruang warna *grayscale* dan CMYK terjadi perubahan jumlah kanal warna. Ruang warna *grayscale* memiliki kanal warna 1 dimensi dengan nilai maksimum 255, sedangkan ruang warna CMYK memiliki 4 kanal warna dengan nilai maksimum 100. Transformasi ruang warna berpengaruh terhadap *feature maps* yang dilakukan oleh proses konvolusi yang terjadi pada model yang dapat dilihat pada Tabel 4.2. Pada ruang warna *grayscale* terjadi penurunan nilai prediksi model. Hal ini terjadi karena ruang warna *grayscale* hanya memiliki 1 kanal warna, sehingga terjadi kehilangan informasi citra akibat transformasi ruang warna. Oleh karena itu *feature maps* yang terbentuk dari hasil konvolusi kurang baik untuk *training*. Semakin banyak konvolusi yang dilakukan, semakin memberikan distorsi ruang, sehingga penggunaan 10 layer konvolusi kurang baik untuk ruang warna *grayscale* akibat adanya kehilangan informasi citra.

Tabel 4. 2 perbandingan nilai akurasi training dengan 15 epoch pada ruang warna keabuan dan CMYK

Ruang Warna		Model yang digunakan		
		3 Layer	7 Layer	10 Layer
	Grayscale	78,01%	42,50%	34,04%
	CMYK	73,75%	76,59%	81,67%

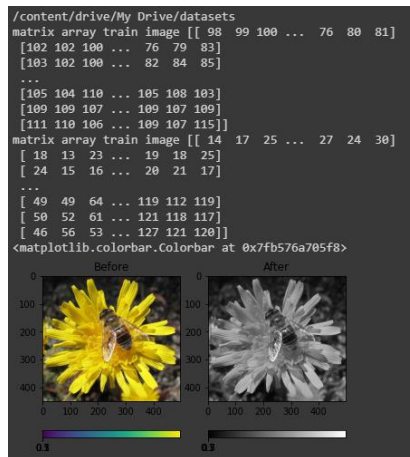
Dari tabel 4.2 diketahui bahwa transformasi ruang warna memengaruhi penggunaan model CNN yang ingin digunakan karena harus memperhatikan hasil transformasi ruang warna terhadap kanal warnanya.

4.2. Transformasi Ruang Warna

Data citra yang telah dikumpulkan dikonversikan pada tiap ruang warna menggunakan modul openCV dengan konversi ruang warna tersebut terjadi transformasi nilai pada tiap kanal warnanya, sehingga tiap ruang warna

memiliki nilainya masing-masing yang memengaruhi proses ekstraksi fitur yang menggunakan CNN.

Tiap ruang warna memiliki ciri khasnya masing-masing. Ruang warna *grayscale* (keabuan) hanya memiliki 1 kanal warna, sedangkan ruang warna RGB, YCbCr, HSV dan XYZ memiliki 3 kanal warna, serta ruang warna CMYK memiliki 4 kanal warna. Oleh karena itu, penentuan penggunaan ruang warna perlu diperhatikan karena tiap ruang warna memiliki ciri masing-masing.



Gambar 4. 1 tampilan array dan konversi citra grayscale

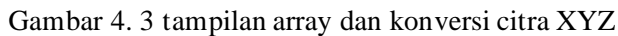
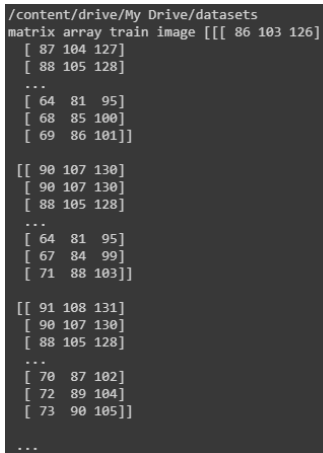
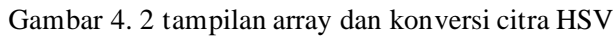
Citra *grayscale* memiliki 1 komponen kanal warna yang menunjukkan warna antara warna putih dan hitam, tidak seperti ruang warna lain yang memiliki banyak komponen warna pada kanal warnanya. Seperti pada ruang warna RGB, HSV, XYZ dan YCbCr yang memiliki 3 kanal warna menjadi citra dengan shape (150,150, 3) atau 3 dimensi, lalu tiap kanal warna tersebut dikonvolusi dengan kernel filter pada tiap layer model CNN sehingga didapatkan ekstraksi fitur yang lebih beragam dibanding ruang warna keabuan yang hanya memiliki 1 kanal warna. Oleh karena itu, ekstraksi fitur yang didapatkan pada ruang *grayscale* kehilangan informasi dari citra yang diberikan sebagai input. Dapat dilihat pada gambar 4.2, gambar 4.3 dan

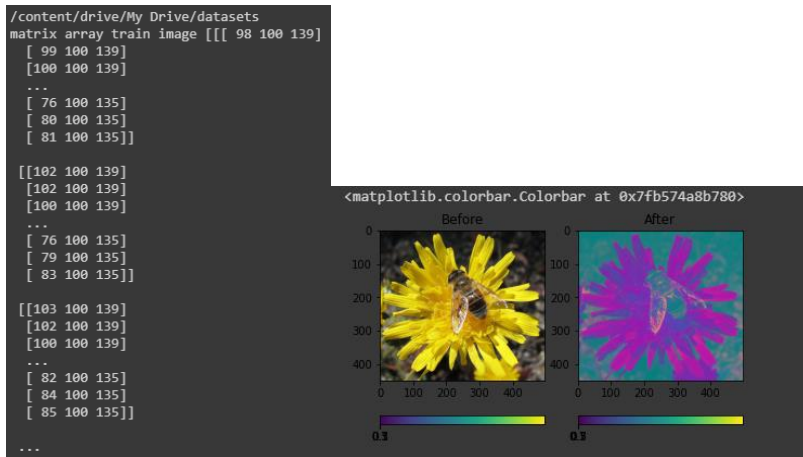
```
content/drive/My Drive/datasets
matrix array train image [[[ 92 126 117]
[ 92 125 118]
[ 92 124 119]
...
[ 87 154 93]
[ 87 147 97]
[ 87 146 98]]]

[[[ 92 122 121]
[ 92 122 121]
[ 92 124 119]
...
[ 87 154 93]
[ 87 149 96]
[ 87 143 100]]]

[[[ 92 121 122]
[ 92 122 121]
[ 92 124 119]
...
[ 87 144 99]
[ 87 141 101]
[ 87 140 102]]]

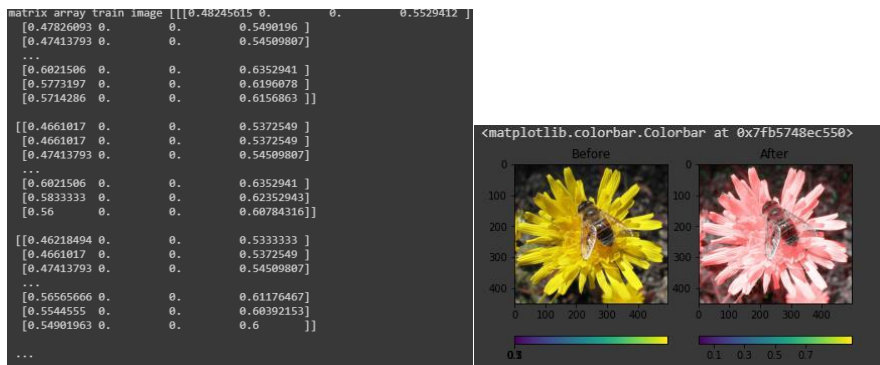
...
```





Gambar 4. 4 tampilan array dan konversi citra YCrCb

Ruang warna CMYK berbeda dengan ruang warna sebelumnya, karena ruang warna CMYK memiliki 4 kanal warna dan memiliki sifat hampir sama dengan RGB. Ruang warna RGB dan CMYK biasa digunakan sebagai dasar pewarnaan digital maupun media cetak. Ruang warna RGB merupakan paduan warna dari ketiga kanal warna dan memberikan warna yang lebih cerah, karena berdasarkan sifat fisika cahaya, tiap penggabungan warna menampilkan warna putih cahaya tersebut. Pada ruang warna CMYK tiap paduan warna memberikan warna yang lebih gelap dan ruang warna ini biasa digunakan untuk sistem mesin cetak.



Gambar 4. 5 tampilan array dan konversi citra CMYK

Ruang warna CMYK memiliki 4 kanal warna. Tidak seperti ruang warna lainnya yang memiliki 3 kanal warna, ruang warna CMYK memiliki *depth* yang lebih banyak. Sehingga fitur yang dihasilkan dari ekstraksi fitur yang dilakukan CNN masih mendapatkan informasi yang tidak jauh berbeda dengan ruang warna RGB.

Untuk memperkuat bukti dari pengaruh ruang warna, digunakan pengulangan proses *training* pada masing-masing ruang warna dengan model CNN-7 *layer* sebanyak 5, lalu data input setiap pengulangan diacak sedemikian rupa secara manual, kemudian dikonversikan lagi ke ruang warna masing-masing. Setelah itu melalui proses training kembali dan hasilnya dibandingkan secara keseluruhan agar dapat dilihat apakah proses yang terjadi tetap sama ketika data input diubah ataupun diacak pada citra ujinya.

Tabel 4. 3 Tabel perbandingan akurasi tiap percobaan pada data yang telah diacak













	percobaan 1	percobaan 2	percobaan 3	percobaan 4	percobaan 5	average
RGB	83,6	88,6	82,9	80,8	82,2	83,62
Gray	42,5	53,2	54,6	47,5	64,6	52,48
HSV	80,1	83,6	63,8	82,9	85,5	79,18
XYZ	76,9	65,2	69,5	72,3	60,9	68,96
CMYK	76,6	85,1	78	77,3	83,7	80,14
YCrCb	58,8	70,9	56	69,3	65,3	64,06







Pada Tabel 4.3 dapat dilihat bahwa ruang warna RGB memiliki rata-rata akurasi terbaik daripada ruang warna lain untuk deteksi kendaraan pada data yang telah disajikan. Dalam hal ini ruang warna RGB untuk kasus *CNN* memang cocok untuk proses *machine learning* dan *deep learning*, karena nilai yang diterima sensor kamera dan yang diolah saat *preprocessing* tidak banyak berubah, berbeda dengan ruang warna lain yang merubah nilai dari tiap kanal warnanya. Ruang warna lain dapat digunakan untuk proses segmentasi citra maupun sistem klasifikasi yang tidak menggunakan proses konvolusi.

4.3. Prediksi tiap ruang warna untuk citra dengan intensitas pencahayaan berbeda

Transformasi kanal warna atau perubahan ruang warna pada obyek yang memiliki warna sekitar dengan kondisi pencahayaan berbeda memberikan hasil yang bervariasi terhadap kondisi citra output tersebut. Dapat dilihat pada tabel 4.4, mobil yang sama dengan pengambilan sudut yang serupa namun dengan kondisi 3 cahaya yang berbeda, membuat hasil yang berbeda-beda pula pada tiap ruang warna yang diuji.

Tabel 4. 4 contoh gambar uji pada tiap ruang warna dengan kondisi cahaya berbeda

	Normal	Gelap	<i>overexposed</i>
RGB			
Grayscale			
HSV			
XYZ			

YCbCr			
CMYK			

Didapatkan hasil pada penggunaan cahaya berbeda yang beragam untuk tiap ruang warna, dapat terlihat bahwa pada Tabel 4.4 penggunaan ruang warna sangat mempengaruhi bentuk dari citra uji tersebut, sehingga ketika dimasukkan pada *Neural Network* terjadi beberapa perubahan prediksi.

Berikut adalah hasil citra uji yang berjumlah 47 diuji sebanyak 5 kali percobaan dengan proses yang sama seperti sebelumnya, yaitu data diacak lalu dikonversi ke tiap ruang warna kemudian *training* dengan masing-masing model, dan model yang digunakan adalah CNN-7 Layer.





Tabel 4. 5 data hasil prediksi dengan citra uji sebanyak 5 kali percobaan

data target	percobaan 1	percobaan 2	percobaan 3	percobaan 4	percobaan 5	average
RGB	63,8	74,4	68,1	70,2	74,4	70,18
Gray	23,4	8,5	6,4	10,6	6,4	11,06
HSV	61,7	61,7	48,9	68,1	76,6	63,4
XYZ	55,3	38,3	48,9	57,4	36,1	47,2
CMYK	53,1	70,2	55,3	72,3	65,9	63,36
YCrCb	14,9	48,9	14,9	31,9	55,3	33,18

Citra yang diuji pada keadaan pencahayaan *overexposed*, seluruh ruang warna tidak dapat memprediksi kendaraan yang diuji, hanya beberapa citra uji yang dapat dinyatakan sebagai mobil. Berbeda dengan keadaan gelap, hampir seluruh ruang warna dapat mengenali sebagai mobil kecuali ruang warna keabuan.

Transformasi ruang warna memberikan pengaruh pada citra karena terjadinya perubahan informasi input oleh citra tersebut, sehingga citra dari tiap ruang warna memberikan informasi yang berbeda ketika di konvolusikan. Proses yang terjadi pada keadaan normal dan minim cahaya masih dapat mengenali nya sebagai objek, karena hasil ekstraksi fitur masih mendapatkan fitur dari kendaraan. Berbeda ketika pencahayaan *over-exposed* yang seluruh ruang warna tidak dapat mengenalnya sebagai fitur kendaraan, hal ini dapat terjadi karena proses konvolusi tidak dapat mendeteksi tepi dari kendaraan akibat dari pencahayaan yang berlebih pada tepi kendaraan citra uji.




Tabel 4. 6 sampel prediksi citra pada ruang warna berbeda di siang hari


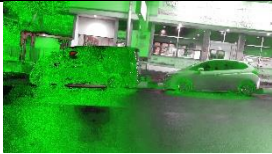

Ruang Warna	Citra	Hasil prediksi
RGB		MOBIL
GrayScale		Bukan mobil
HSV		Mobil
XYZ		Mobil

CMYK		Mobil
YCrCB		Mobil

Pada Tabel 4.6 dan Tabel 4.7 memiliki pola yang sama dengan Tabel 4.4, dimana hasil ekstraksi fitur dapat dikenali sebagai fitur kendaraan dengan baik. Sama halnya yang terjadi pada keadaan pencahayaan malam hari dimana ruang warna RGB, HSV dan CMYK dapat mengenali hasil ekstraksi fitur sebagai fitur dari kendaraan.

Tabel 4. 7 Sampel prediksi pada kondisi malam hari

Ruang Warna	Citra	Hasil Prediksi
RGB		mobil
Grayscale		Bukan mobil
HSV		Mobil

XYZ		Bukan mobil
CMYK		MOBIL
YCrCb		Bukan mobil

Ada beberapa ruang warna yang kurang cocok digunakan untuk keadaan gelap dan keadaan terang, dan ada juga ruang warna yang cocok di keduanya. Dapat dilihat pada tabel 4.6 bahwa ruang warna RGB dan CMYK memiliki akurasi yang sama tingginya dibandingkan dengan ruang warna lainnya.

Tabel 4. 8 perbandingan nilai akurasi citra uji

	RGB	Gray	HSV	XYZ	CMYK	YCrCb	Total Posit
True Posit	23	1	16	11	23	10	24
Akurasi	95.83333	4.166667	66.66667	45.83333	95.83333	41.66667	

Dapat dilihat dari tabel 4.5 dan 4.8 bahwa ruang warna RGB dan CMYK dapat menjadi ruang warna untuk mendeteksi kendaraan pada posisi pencahayaan minim maupun optimal, namun tidak dapat mendeteksi kendaraan dengan pencahayaan yang berlebihan. Hal ini terjadi karena pencahayaan yang berlebih menutupi sisi obyek dari citra tersebut, sehingga sisi ataupun *object interest* tidak tampak dan tertutupi oleh cahaya yang berlebih.

BAB V

PENUTUP

5.1 KESIMPULAN

Telah dilakukan penelitian untuk mengetahui pengaruh transformasi kanal warna terhadap ekstraksi fitur untuk deteksi obyek yang dapat disimpulkan bahwa:

1. Penggunaan model CNN dengan 7-layer konvolusi terbukti mendapatkan hasil akurasi dengan nilai akurasi hingga 88%
2. Transformasi kanal warna yang terjadi mempengaruhi akurasi akibat adanya perubahan jumlah kanal warna yang terjadi, sehingga terjadi kurangnya informasi citra, seperti pada ruang warna grayscale, YCbCr dan XYZ.
3. Ruang warna terbaik untuk ekstraksi fitur yaitu ruang warna RGB, CMYK dan HSV dengan nilai akurasi ~80%, ruang warna tersebut dapat mengekstraksi fitur pada citra pencahayaan normal dan minim cahaya.

5.2 SARAN

Pada penelitian selanjutnya dapat dilakukan dengan saran berikut ini guna meningkatkan penelitian dengan lebih akurat dan presisi:

1. Penambahan jumlah data set agar proses *training* dapat lebih optimal
2. Penggunaan *image augmentation* ketika terjadi *over-fit* /*under-fit* pada grafik *training*
3. Proses transformasi kanal warna perlu dioptimalkan karena memakan waktu yang lama

DAFTAR PUSTAKA

1. Shapiro, L.G. & Stockman, G. C. (2001). *Computer Vision*. New Jersey: Prentice Hall.
2. Lazaro, A., Buliali, J. L., & Amaliah, B. (2017). Deteksi Kecepatan Kendaraan Berjalan di Jalan Menggunakan OpenCV. *Jurnal Teknik ITS*, 6(2). <https://doi.org/10.12962/j23373539.v6i2.23489>
3. Gowda, S. N., & Yuan, C. (2019). ColorNet: Investigating the Importance of Color Spaces for Image Classification. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11364 LNCS, 581–596. https://doi.org/10.1007/978-3-030-20870-7_36
4. Sonka, M. et al. (2008). *Image processing, analysis and machine vision. Neurocomputing* (Vol. 6). Iowa: Cengage Learning. [https://doi.org/10.1016/0925-2312\(94\)90073-6](https://doi.org/10.1016/0925-2312(94)90073-6)
5. David A, Forsyth. Jean, P. (2003). *Computer Vision : A Modern Approach*. New Jersey: Pearson Education.
6. Richard, S. (2011). *Computer vision. Computer Vision: Algorithms and Applications*. London: Springer. <https://doi.org/10.4324/9780429042522-10>
7. Sutoyo, T. Mulyanto, E. Suhartono, V. Nurhayati, O. (2009). *Teori Pengolahan citra digital*. Yogyakarta: Andi.
8. Putra, D. (2010). *Pengolahan Citra Digital - Darma Putra - Google Buku*. Yogyakarta: Andi.
9. Wyszecky, Guthrie. Stiles, W. . (1982). *Color Science: Concepts and Methods, Quantitative Data and Formulae, 2nd ed*. New York: John Wiley and Sons. <https://doi.org/10.1002/ieam.1649>
10. Hunt, R. W. G. (2005). *The reproduction of colour. Color Research & Application* (2nd ed., Vol. 30). <https://doi.org/10.1002/col.20163>
11. Kadir, A., & Susanto, A. (2013). *Teori dan Aplikasi Pengolahan*

Citra. Yogyakarta: Penerbit Andi.

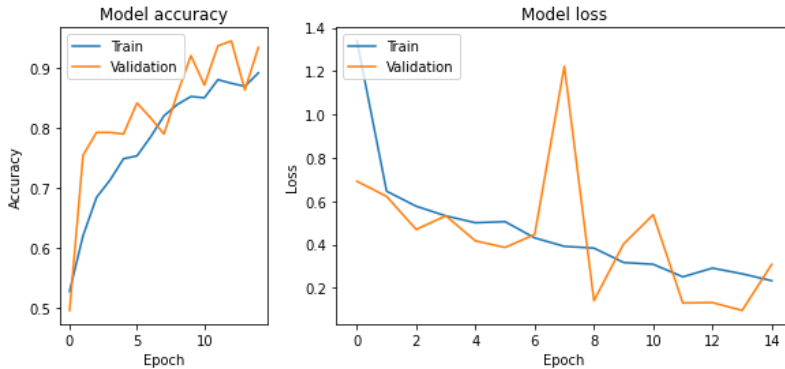
12. Gonzalez, R. C., Woods, R. E., & Masters, B. R. (2002). *Digital Image Processing, Third Edition*. New Jersey: Pearson Prentice Hall. <https://doi.org/10.1117/1.3115362>
13. Inc., M. (2020). Convert Between RGB and HSV Color Spaces - MATLAB & Simulink. Retrieved February 27, 2020, from <https://www.mathworks.com/help/images/convert-from-hsv-to-rgb-color-space.html>
14. Ford, A., & Adrian, R. (1998). Colour Space Conversions, 1998, 1–31.
15. Cuturicu, C. (1999). A note about the JPEG decoding algorithm. Retrieved February 27, 2020, from www.inforamp.net/~poynton/PDFs/coloureq.pdf.
16. Russell, S., & Norvig, P. (2010). *Artificial Intelligence A Modern Approach Third Edition*. Pearson. New Jersey: Pearson Education. <https://doi.org/10.1017/S0269888900007724>
17. Shwartz, S. S., & David, S. B. (2014). *Understanding machine learning: From theory to algorithms. Understanding Machine Learning: From Theory to Algorithms* (Vol. 9781107057). Cambridge: Cambridge University Press. <https://doi.org/10.1017/CBO9781107298019>
18. Morgan, P. (2018). *Machine Learning Is Changing the Rules*. Sebastapol: O'Reilly Media. <https://doi.org/10.1016/j.transproceed.2007.01.092>
19. Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
20. Haykin, S. (2009). *Neural networks and learning. Institute of Physics Conference Series* (Vol. 127). Hamilton: Pearson Prentice Hall.
21. Suyanto. (2014). *Artificial Intelligence*. Bandung: INFORMATIKA.

22. Clause, Führer. Solem, Jan Erik. verdier, olivier. (2016). *Scientific Computing with Python 3* (2nd ed.). Birmingham: Packt Publishing.
23. Abadi, M., Isard, M., & Murray, D. G. (2017). A computational model for TensorFlow an introduction. *MAPL 2017 - Proceedings of the 1st ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, co-located with PLDI 2017*, 1–7. <https://doi.org/10.1145/3088525.3088527>
24. Brownlee, J. (2016). Introduction to Python Deep Learning with Keras. Retrieved January 27, 2020, from <https://machinelearningmastery.com/introduction-python-deep-learning-library-keras/>

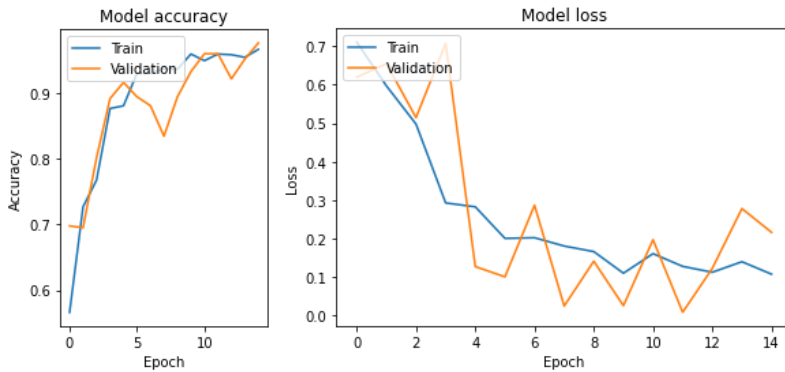
LAMPIRAN A DATA HASIL PENELITIAN

1. Hasil Training model berdasarkan ruang warna

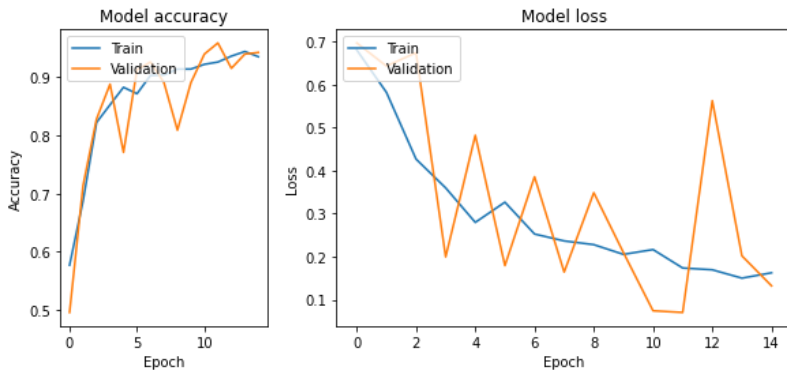
A. Ruang warna RGB



gambar a11. Hasil training model CNN 3 Layer ruang warna RGB

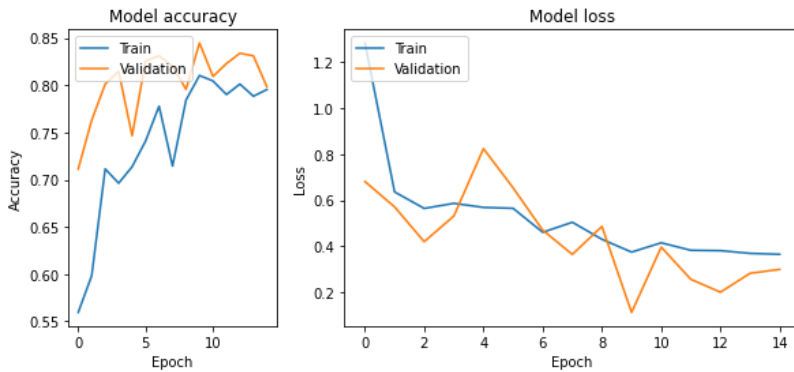


Gambar a12. Hasil training model CNN 7 Layer ruang warna RGB

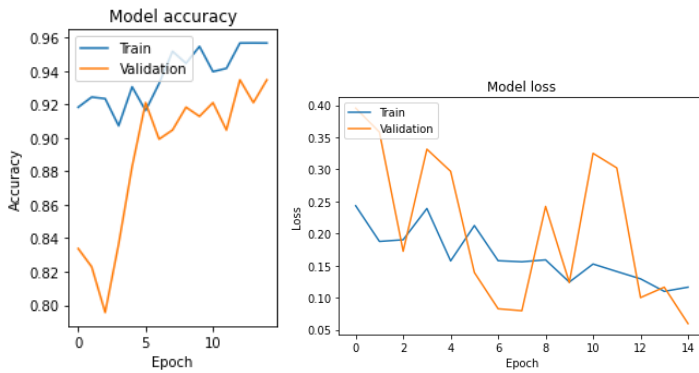


Gambar a13. Hasil training model CNN 10 Layer ruang warna RGB

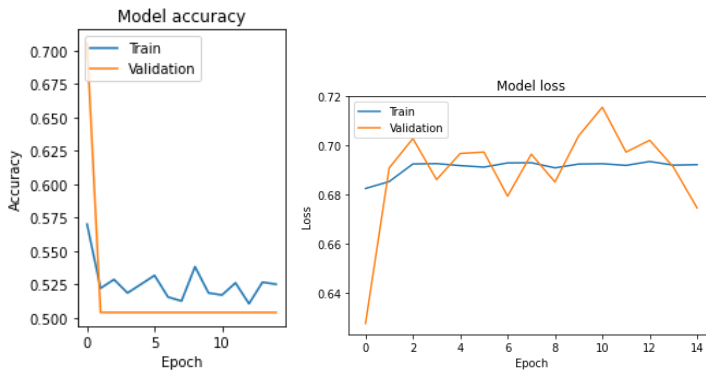
B. Ruang warna Grayscale



Gambar b11. Hasil training model CNN 3 Layer ruang warna Grayscale

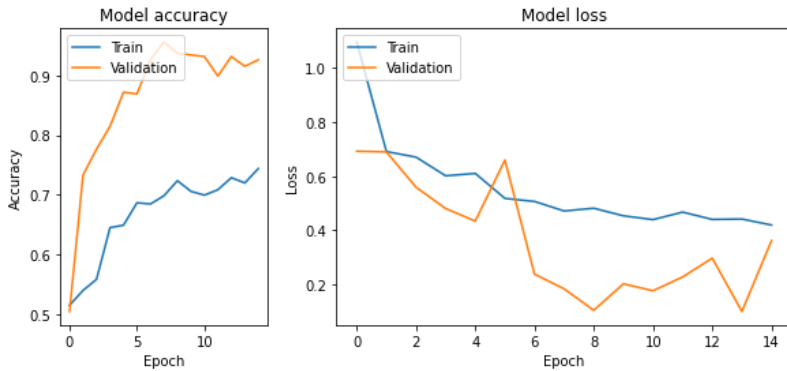


Gambar b12. Hasil training model CNN 7 Layer ruang warna Grayscale

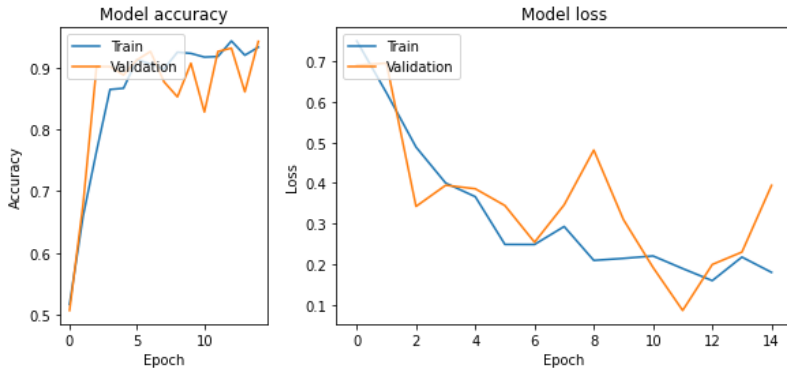


Gambar b13. Hasil training model CNN 10 Layer ruang warna Grayscale

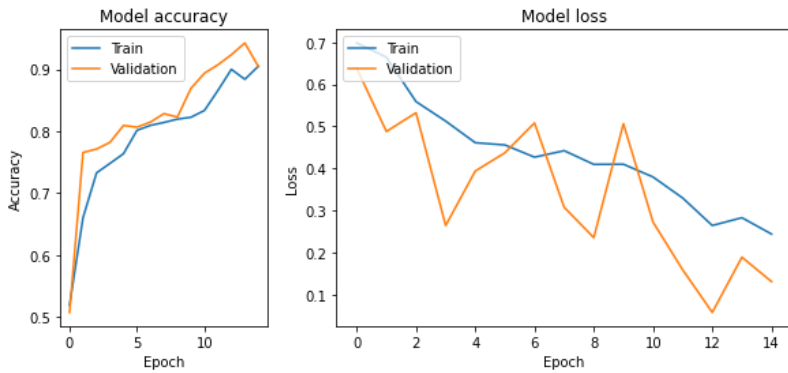
C. Ruang warna HSV



gambar c11. Hasil Training model CNN 3 Layer ruang warna HSV

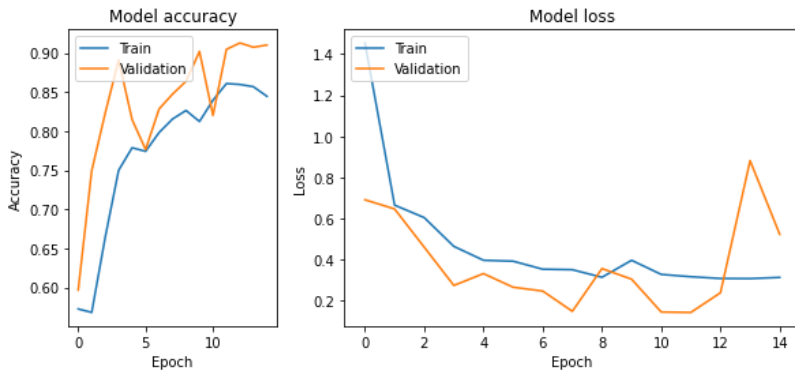


gambar c12. Hasil training model CNN 7 Layer ruang warna HSV

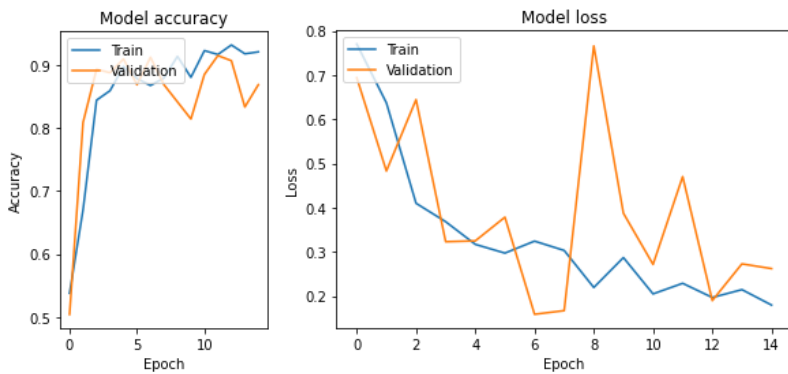


gambar c13. Hasil training model CNN 10 Layer ruang warna HSV

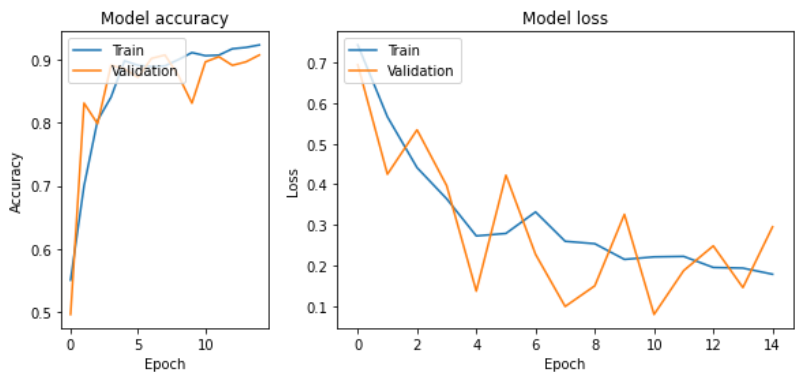
D. Ruang warna CMYK



gambar d11. Hasil Training model CNN 3 Layer ruang warna CMYK

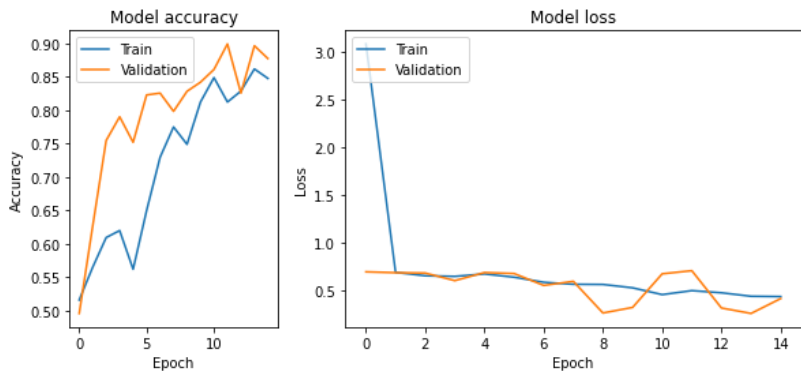


Gambar d12. Hasil training model CNN 7 Layer ruang warna CMYK

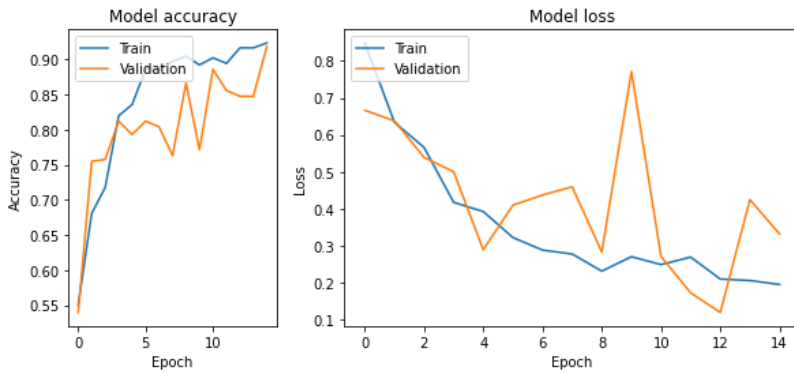


Gambar d13. Hasil training model CNN 10 Layer ruang warna CMYK

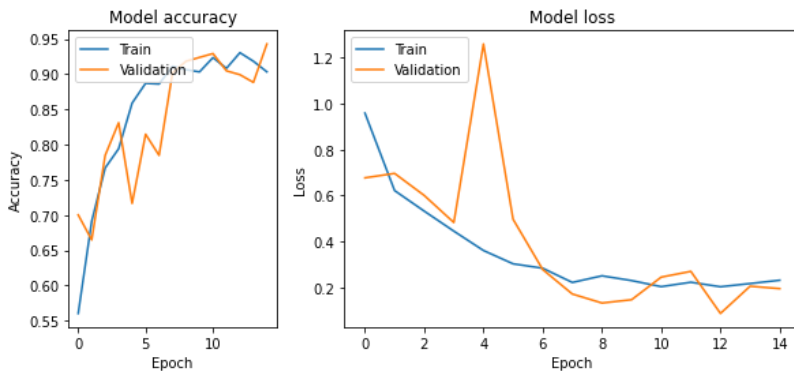
E. Ruang warna XYZ



gambar e11. Hasil Training model CNN 3 Layer ruang warna XYZ

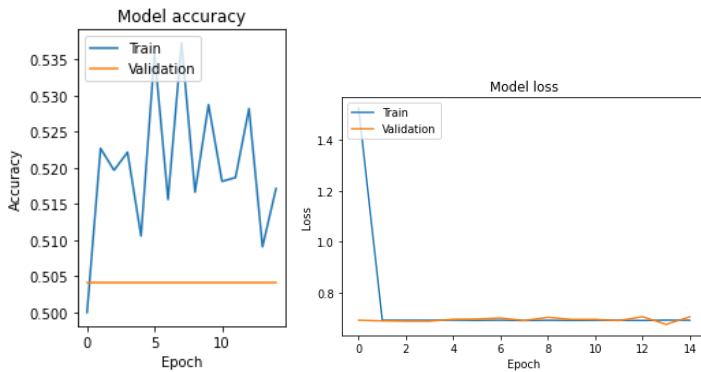


Gambar e12. Hasil Training model CNN 7 Layer ruang warna XYZ

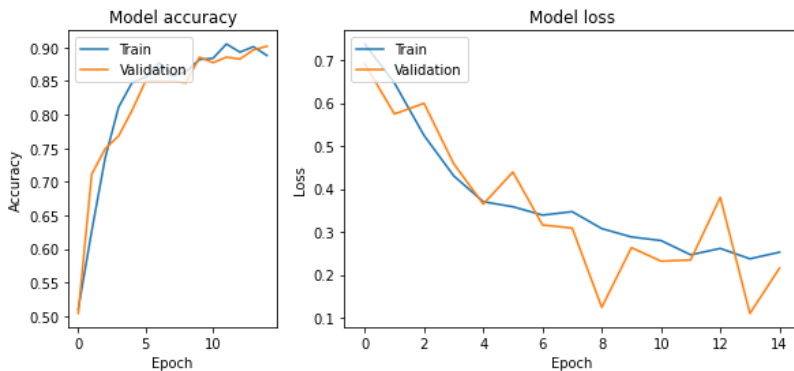


Gambar e13. Hasil Training model CNN 10 Layer ruang warna XYZ

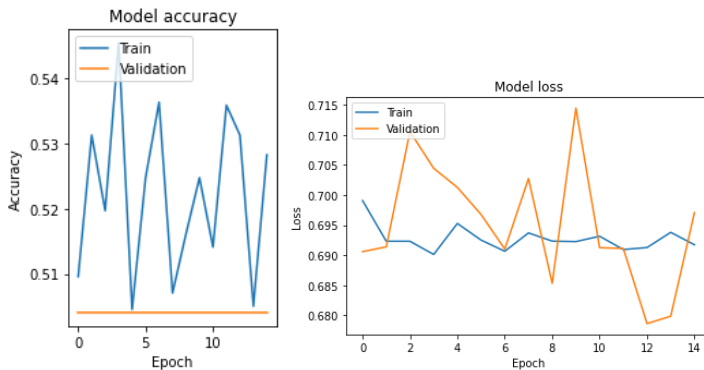
F. Ruang warna YcrCb



Gambar f11. Hasil Training model CNN 3 Layer ruang warna YCrCb



Gambar f12. Hasil Training model CNN 7 Layer ruang warna YCrCb



Gambar f13. Hasil training model CNN 10 Layer ruang warna YcrCb

2. Confusion Matrix tiap ruang warna

2.1 percobaan 1

Tabel 2.1.A Confusion Matrix ruang warna RGB

	RGB			
Actual\Predict	Positif	Negatif	Total Positif	81
Positif	61	20	Total Negatif	60
Negatif	3	57	Total	141
Accuracy	0,836879	Specificity	0,7125	
error rate	0,163121	Precision	0,953125	
Sensitivity	0,753086	False Positif	0,05	

Tabel 2.1.B Confusion Matrix ruang warna Grayscale

	Grayscale			
Actual\Predict	Positif	Negatif	Total Positif	81
Positif	40	41	Total Negatif	60
Negatif	40	20	Total	141
Accuracy	0,425532	Specificity	0,19802	
error rate	0,574468	Precision	0,5	
Sensitivity	0,493827	False Positif	0,666667	

Tabel 2.1.C Confusion Matrix ruang warna HSV

	HSV			
Actual\Pred	Positif	Negatif	Total Positif	81
Positif	58	23	Total Negatif	60
Negatif	5	55	Total	141
Accuracy	0,801418	Specificity	0,662651	
error rate	0,198582	Precision	0,920635	
Sensitivity	0,716049	False Positif	0,083333	

Tabel 2.1.D Confusion Matrix ruang warna XYZ

	XYZ			
Actual\Pred	Positif	Negatif	Total Positif	81
Positif	57	24	Total Negatif	60
Negatif	13	47	Total	141
Accuracy	0,737589	Specificity	0,559524	
error rate	0,262411	Precision	0,814286	
Sensitivity	0,703704	False Positif	0,216667	

Tabel 2.1.E Confusion Matrix ruang warna CMYK

	CMYK			
Actual\Pred	Positif	Negatif	Total Positif	81
Positif	55	26	Total Negatif	60
Negatif	7	53	Total	141
Accuracy	0,765957	Specificity	0,616279	
error rate	0,234043	Precision	0,887097	
Sensitivity	0,679012	False Positif	0,116667	

Tabel 2.1.F Confusion Matrix ruang warna YCrCb

	YCrCb			
Actual\Predict	Positif	Negatif	Total Positif	81
Positif	36	45	Total Negatif	60
Negatif	13	47	Total	141
Accuracy	0,588652	Specificity	0,447619	
error rate	0,411348	Precision	0,734694	
Sensitivity	0,444444	False Positive	0,216667	

2.2 Percobaan 2

Tabel 2.2.A Confusion Matrix ruang warna RGB

	RGB			
Actual\Predict	Positif	Negatif	Total Positif	81
Positif	68	13	Total Negatif	60
Negatif	3	57	Total	141
Accuracy	0,88652482	Specificity	0,78082192	
error rate	0,11347518	Precision	0,95774648	
Sensitivity	0,83950617	False Positive	0,05	

Tabel 2.2.B Confusion Matrix ruang warna Grayscale

	Grayscale			
Actual\Predict	Positif	Negatif	Total Positif	81
Positif	33	48	Total Negatif	60
Negatif	18	42	Total	141
Accuracy	0,53191489	Specificity	0,38888889	
error rate	0,46808511	Precision	0,64705882	
Sensitivity	0,40740741	False Positive	0,3	

Tabel 2.2.C Confusion Matrix ruang warna HSV

	HSV			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	61	20	Total Negatif	60
Negatif	3	57	Total	141
Accuracy	0,83687943	Specificity	0,7125	
error rate	0,16312057	Precision	0,953125	
Sensitivity	0,75308642	False Positive	0,05	

Tabel 2.2.D Confusion Matrix ruang warna XYZ

	XYZ			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	50	31	Total Negatif	60
Negatif	18	42	Total	141
Accuracy	0,65248227	Specificity	0,46153846	
error rate	0,34751773	Precision	0,73529412	
Sensitivity	0,61728395	False Positive	0,3	

Tabel 2.2.E Confusion Matrix ruang warna CMYK

	CMYK			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	64	17	Total Negatif	60
Negatif	4	56	Total	141
Accuracy	0,85106383	Specificity	0,72727273	
error rate	0,14893617	Precision	0,94117647	
Sensitivity	0,79012346	False Positive	0,06666667	

Tabel 2.2.F Confusion Matrix ruang warna YCbCr

	YCrCb			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	55	26	Total Negatif	60
Negatif	15	45	Total	141
Accuracy	0,70921986	Specificity	0,52325581	
error rate	0,29078014	Precision	0,78571429	
Sensitivity	0,67901235	False Positive	0,25	

2.3 Percobaan 3

Tabel 2.3.A Confusion Matrix ruang warna RGB

	RGB			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	62	19	Total Negatif	60
Negatif	5	55	Total	141
Accuracy	0,82978723	Specificity	0,69620253	
error rate	0,17021277	Precision	0,92537313	
Sensitivity	0,7654321	False Positive	0,08333333	

Tabel 2.3.B Confusion Matrix ruang warna Grayscale

	Grayscale			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	34	47	Total Negatif	60
Negatif	17	43	Total	141
Accuracy	0,54609929	Specificity	0,40186916	
error rate	0,45390071	Precision	0,66666667	
Sensitivity	0,41975309	False Positive	0,28333333	

Tabel 2.3.C Confusion Matrix ruang warna HSV

	HSV			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	53	28	Total Negatif	60
Negatif	23	37	Total	141
Accuracy	0,63829787	Specificity	0,42045455	
error rate	0,36170213	Precision	0,69736842	
Sensitivity	0,65432099	False Positive	0,38333333	

Tabel 2.3.D Confusion Matrix ruang warna XYZ

	XYZ			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	56	25	Total Negatif	60
Negatif	18	42	Total	141
Accuracy	0,69503546	Specificity	0,49411765	
error rate	0,30496454	Precision	0,75675676	
Sensitivity	0,69135802	False Positive	0,3	

Tabel 2.3.E Confusion Matrix ruang warna CMYK

	CMYK			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	57	24	Total Negatif	60
Negatif	7	53	Total	141
Accuracy	0,78014184	Specificity	0,63095238	
error rate	0,21985816	Precision	0,890625	
Sensitivity	0,7037037	False Positive	0,11666667	

Tabel 2.3.F Confusion Matrix ruang warna YCbCr

	YCrCb			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	32	49	Total Negatif	60
Negatif	13	47	Total	141
Accuracy	0,56028369	Specificity	0,43119266	
error rate	0,43971631	Precision	0,71111111	
Sensitivity	0,39506173	False Positive	0,21666667	

2.4 Percobaan 4

2.4.A Confusion Matrix ruang Warna RGB

	RGB			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	66	15	Total Negatif	60
Negatif	12	48	Total	141
Accuracy	0,80851064	Specificity	0,64	
error rate	0,19148936	Precision	0,84615385	
Sensitivity	0,81481481	False Positive	0,2	

2.4.B Confusion Matrix ruang Warna Grayscale

	Grayscale			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	34	47	Total Negatif	60
Negatif	27	33	Total	141
Accuracy	0,4751773	Specificity	0,30841121	
error rate	0,5248227	Precision	0,55737705	
Sensitivity	0,41975309	False Positive	0,45	

2.4.C Confusion Matrix ruang Warna HSV

	HSV			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	62	19	Total Negatif	60
Negatif	5	55	Total	141
Accuracy	0,82978723	Specificity	0,69620253	
error rate	0,17021277	Precision	0,92537313	
Sensitivity	0,7654321	False Positive	0,08333333	

2.4.D Confusion Matrix ruang Warna XYZ

	XYZ			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	59	22	Total Negatif	60
Negatif	17	43	Total	141
Accuracy	0,72340426	Specificity	0,52439024	
error rate	0,27659574	Precision	0,77631579	
Sensitivity	0,72839506	False Positive	0,28333333	

2.4.E Confusion Matrix ruang Warna CMYK

	CMYK			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	53	28	Total Negatif	60
Negatif	4	56	Total	141
Accuracy	0,77304965	Specificity	0,63636364	
error rate	0,22695035	Precision	0,92982456	
Sensitivity	0,65432099	False Positive	0,06666667	

2.4.F Confusion Matrix ruang Warna YCbCr

	YCrCb			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	38	43	Total Negatif	60
Negatif	13	47	Total	141
Accuracy	0,60283688	Specificity	0,45631068	
error rate	0,39716312	Precision	0,74509804	
Sensitivity	0,4691358	False Positive	0,21666667	

2.5 Percobaan 5

2.5.A Confusion Matrix ruang Warna RGB

	RGB			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	66	15	Total Negatif	60
Negatif	10	50	Total	141
Accuracy	0,82269504	Specificity	0,66666667	
error rate	0,17730496	Precision	0,86842105	
Sensitivity	0,81481481	False Positive	0,16666667	

2.5.B Confusion Matrix ruang Warna Grayscale

	Grayscale			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	63	18	Total Negatif	60
Negatif	32	28	Total	141
Accuracy	0,64539007	Specificity	0,35897436	
error rate	0,35460993	Precision	0,66315789	
Sensitivity	0,77777778	False Positive	0,53333333	

2.5.C Confusion Matrix ruang Warna HSV

	HSV			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	60	11	Total Negatif	60
Negatif	8	52	Total	131
Accuracy	0,85496183	Specificity	0,73239437	
error rate	0,14503817	Precision	0,88235294	
Sensitivity	0,74074074	False Positive	0,13333333	

2.5.D Confusion Matrix ruang Warna XYZ

	XYZ			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	49	32	Total Negatif	60
Negatif	23	37	Total	141
Accuracy	0,60992908	Specificity	0,40217391	
error rate	0,39007092	Precision	0,68055556	
Sensitivity	0,60493827	False Positive	0,38333333	

2.5.E Confusion Matrix ruang Warna CMYK

	CMYK			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	63	18	Total Negatif	60
Negatif	5	55	Total	141
Accuracy	0,83687943	Specificity	0,70512821	
error rate	0,16312057	Precision	0,92647059	
Sensitivity	0,77777778	False Positive	0,08333333	

2.5.F Confusion Matrix ruang Warna YCbCr

	YCrCb			
Actual\Predic	Positif	Negatif	Total Positif	81
Positif	55	26	Total Negatif	60
Negatif	22	38	Total	141
Accuracy	0,65957447	Specificity	0,44186047	
error rate	0,34042553	Precision	0,71428571	
Sensitivity	0,67901235	False Positive	0,36666667	

LAMPIRAN B

PROGRAM

Mount Drive Google

```
from google.colab import drive
drive.mount('/content/drive')
```

Create Dataset Folder

```
import os

directory = '/content/drive/My Drive/datasets/'
directory_model = '/content/drive/My Drive/model/'
rgb = directory+'rgb'
grayscale = directory+'grayscale'
hsv = directory+'hsv'
cmy = directory+'cmy'
ycbcr = directory+'ycbcr'
xyz = directory+'xyz'

os.getcwd()
os.chdir(directory)
!pwd
try:
    if not os.path.exists(grayscale):
        !mkdir grayscale
        print("directory is ready!")
    if not os.path.exists(hsv):
        !mkdir hsv
        print("directory is ready!")
    if not os.path.exists(cmy):
        !mkdir cmy
        print("directory is ready!")
    if not os.path.exists(ycbcr):
        !mkdir ycbcr
        print("directory is ready!")
    if not os.path.exists(xyz):
        !mkdir xyz
        print("directory is ready!")
    if not os.path.exists(rgb):
        !mkdir rgb
        print("directory is ready!")

except OSError:
    print ('Error: Creating directory. ')

os.getcwd()
os.chdir(directory_model)
!pwd
```

```

rgb_ = directory_model+'rgb'
grayscale_ = directory+'grayscale'
hsv_ = directory_model+'hsv'
cmv_ = directory_model+'cmv'
ybcbr_ = directory_model+'ybcbr'
xyz_ = directory_model+'xyz'
try:
    if not os.path.exists(rgb_):
        !mkdir rgb
        print("directory is ready!")
    if not os.path.exists(grayscale_):
        !mkdir grayscale
        print("directory is ready!")
    if not os.path.exists(hsv_):
        !mkdir hsv
        print("directory is ready!")
    if not os.path.exists(cmv_):
        !mkdir cmv
        print("directory is ready!")
    if not os.path.exists(ybcbr_):
        !mkdir ybcbr
        print("directory is ready!")
    if not os.path.exists(xyz_):
        !mkdir xyz
        print("directory is ready!")

except OSError:
    print ('Error: Creating directory. ')
def create_folder_test():
    try:
        if not os.path.exists(test):
            !mkdir test
            print("directory is ready!")
        if not os.path.exists(train):
            !mkdir train
            print("directory is ready!")
        if not os.path.exists(test_image):
            !mkdir test_image
            print("directory is ready!")
    except OSError:
        print ('Error: Creating directory. ')

def create_folder_mobil_bukan_mobil():
    try:
        if not os.path.exists(mobil):
            !mkdir mobil
            print("directory is ready!")
        if not os.path.exists(bukan_mobil):
            !mkdir bukan_mobil
            print("directory is ready!")
    except OSError:
        print ('Error: Creating directory. ')

```

```

##### GRAYSCALE ###
os.getcwd()
os.chdir( grayscale )
!pwd

test = grayscale+'/test'
train = grayscale+'/train'
test_image = grayscale+'test_image'
mobil = grayscale+'/mobil'
bukan_mobil = grayscale+'/bukan_mobil'

create_folder_test()

os.getcwd()
os.chdir( grayscale+'/test' )
!pwd
create_folder_mobil_bukan_mobil()
os.getcwd()
os.chdir( grayscale+'/train' )
!pwd
create_folder_mobil_bukan_mobil()
##### HSV #####
os.getcwd()
os.chdir( hsv )
!pwd

test = hsv+'/test'
train = hsv+'/train'
test_image = hsv+'test_image'
mobil = hsv+'/mobil'
bukan_mobil = hsv+'/bukan_mobil'

create_folder_test()

os.getcwd()
os.chdir( hsv+'/test' )
!pwd
create_folder_mobil_bukan_mobil()
os.getcwd()
os.chdir( hsv+'/train' )
!pwd
create_folder_mobil_bukan_mobil()
##### CMY #####
os.getcwd()
os.chdir( cmy )
!pwd

test = cmy+'/test'
train = cmy+'/train'
test_image = cmy+'test_image'
mobil = cmy+'/mobil'
bukan_mobil = cmy+'/bukan_mobil'

```

```

create_folder_test()

os.getcwd()
os.chdir(cmy+'/test')
!pwd
create_folder_mobil_bukan_mobil()
os.getcwd()
os.chdir(cmy+'/train')
!pwd
create_folder_mobil_bukan_mobil()
##### YCBR #####
os.getcwd()
os.chdir(ycbcr)
!pwd

test = ycbcr+'/test'
train = ycbcr+'/train'
test_image = ycbcr+'test_image'
mobil = ycbcr+'/mobil'
bukan_mobil = ycbcr+'/bukan_mobil'

create_folder_test()

os.getcwd()
os.chdir(ycbcr+'/test')
!pwd
create_folder_mobil_bukan_mobil()
os.getcwd()
os.chdir(ycbcr+'/train')
!pwd
create_folder_mobil_bukan_mobil()
##### XYZ #####
os.getcwd()
os.chdir(xyz)
!pwd

test = xyz+'/test'
train = xyz+'/train'
test_image = xyz+'test_image'
mobil = hsv+'/mobil'
bukan_mobil = hsv+'/bukan_mobil'

create_folder_test()

os.getcwd()
os.chdir(xyz+'/test')
!pwd
create_folder_mobil_bukan_mobil()
os.getcwd()
os.chdir(xyz+'/train')
!pwd
create_folder_mobil_bukan_mobil()

```

Import modul Computer Vision

```

import cv2
import os
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

```

Converting to Grayscale Color Space

```

os.getcwd()
os.chdir(directory)
!pwd

w=10
h=10

####source data####
test = rgb+'/test/'
train = rgb+'/train/'
test_image = rgb+'/test_image/'

####destination data####
test_grayscale = grayscale+'/test/'
train_grayscale = grayscale+'/train/'
test_image_grayscale = grayscale+'/test_image/'

def rgb_to_grayscale(image):
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    return(gray_image)

for file_test_image in os.listdir(test_image):
    img = mpimg.imread(test_image+file_test_image)
    gray = rgb_to_grayscale(img)
    cv2.imwrite(test_image_grayscale+file_test_image,gray)

for folder in os.listdir(test):
    for file_ in os.listdir(test+folder):
        img = mpimg.imread(test+folder+'/'+file_)
        gray = rgb_to_grayscale(img)
        cv2.imwrite(test_grayscale+folder+'/'+file_,gray)

for folder in os.listdir(train):
    for file_ in os.listdir(train+folder):
        img = mpimg.imread(train+folder+'/'+file_)
        gray = rgb_to_grayscale(img)
        cv2.imwrite(train_grayscale+folder+'/'+file_,gray)
        # print(test_grayscale+folder+'/'+file_)
        print('matrix array train image',gray)

fig = plt.figure()
a = fig.add_subplot(1, 2, 1)
imgplot = plt.imshow(img)
a.set_title('Before')

```

```
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
a = fig.add_subplot(1, 2, 2)
imgplot = plt.imshow(gray,cmap='gray', vmin = 0, vmax = 255)
a.set_title('After')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
```

Converting to HSV Colorspace

```
os.getcwd()
os.chdir(directory)
!pwd

####source data####
test = rgb+'/test/'
train = rgb+'/train/'
test_image = rgb+'/test_image/'

####destination data####
test_hsv = hsv+'/test/'
train_hsv = hsv+'/train/'
test_image_hsv = hsv+'/test_image/'

def rgb_to_hsv(image):
    hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    return(hsv_image)

for file_test_image in os.listdir(test_image):
    img = mpimg.imread(test_image+file_test_image)
    hsv = rgb_to_hsv(img)
    cv2.imwrite(test_image_hsv+file_test_image,hsv)

for folder in os.listdir(test):
    for file_ in os.listdir(test+folder):
        img = mpimg.imread(test+folder+'/'+file_)
        hsv = rgb_to_hsv(img)
        cv2.imwrite(test_hsv+folder+'/'+file_,hsv)

for folder in os.listdir(train):
    for file_ in os.listdir(train+folder):
        img = mpimg.imread(train+folder+'/'+file_)
        hsv = rgb_to_hsv(img)
        cv2.imwrite(train_hsv+folder+'/'+file_,hsv)
        # print(test_hsv+folder+'/'+file_)
    print('matrix array train image',hsv)

fig = plt.figure()
a = fig.add_subplot(1, 2, 1)
imgplot = plt.imshow(img)
```

```

a.set_title('Before')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
a = fig.add_subplot(1, 2, 2)
imgplot = plt.imshow(hsv, cmap='hsv', vmin = 0, vmax = 255)
a.set_title('After')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')

```

Converting to XYZ Color Space

```

os.getcwd()
os.chdir(directory)
!pwd

from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

#####source data#####
test = rgb+'/test/'
train = rgb+'/train/'
test_image = rgb+'/test_image/'

#####destination data#####
test_xyz = '/content/drive/My Drive/datasets/xyz/test/'
train_xyz = '/content/drive/My Drive/datasets/xyz/train/'
test_image_xyz = '/content/drive/My Drive/datasets/xyz/test_image/'

def rgb_to_xyz(image):
    xyz_image = cv2.cvtColor(image, cv2.COLOR_BGR2XYZ)
    return(xyz_image)

for file_test_image in os.listdir(test_image):
    img = mpimg.imread(test_image+file_test_image)
    xyz = rgb_to_xyz(img)
    cv2.imwrite(test_image_xyz+file_test_image,xyz)

for folder in os.listdir(test):
    for file_ in os.listdir(test+folder):
        img = mpimg.imread(test+folder+'/'+file_)
        xyz = rgb_to_xyz(img)
        cv2.imwrite(test_xyz+folder+'/'+file_,xyz)

for folder in os.listdir(train):
    for file_ in os.listdir(train+folder):
        img = mpimg.imread(train+folder+'/'+file_)
        xyz = rgb_to_xyz(img)
        cv2.imwrite(train_xyz+folder+'/'+file_,xyz)
        # print(test_xyz+folder+'/'+file_)
    print('matrix array train image',xyz)

fig = plt.figure()
a = fig.add_subplot(1, 2, 1)
imgplot = plt.imshow(img)
a.set_title('Before')

```



```
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
a = fig.add_subplot(1, 2, 2)
imgplot = plt.imshow(xyz)
a.set_title('After')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
```

Converting to YCrCb Color Space

```
os.getcwd()
os.chdir(directory)
!pwd

from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

#####source data#####
test = rgb+'/test/'
train = rgb+'/train/'
test_image = rgb+'/test_image/'

#####destination data#####
test_ycbcr = '/content/drive/My Drive/datasets/ycbcr/test/'
train_ycbcr = '/content/drive/My Drive/datasets/ycbcr/train/'
test_image_ycbcr = '/content/drive/My Drive/datasets/ycbcr/test_image/'

def rgb_to_ycbcr(image):
    hsv_image = cv2.cvtColor(image, cv2.COLOR_BGR2YCrCb)
    return(hsv_image)

for file_test_image in os.listdir(test_image):
    img = mpimg.imread(test_image+file_test_image)
    ycbcr = rgb_to_ycbcr(img)
    cv2.imwrite(test_image_ycbcr+file_test_image,ycbcr)

for folder in os.listdir(test):
    for file_ in os.listdir(test+folder):
        img = mpimg.imread(test+folder+'/'+file_)
        ycbcr = rgb_to_ycbcr(img)
        cv2.imwrite(test_ycbcr+folder+'/'+file_,ycbcr)

for folder in os.listdir(train):
    for file_ in os.listdir(train+folder):
        img = mpimg.imread(train+folder+'/'+file_)
        ycbcr = rgb_to_ycbcr(img)
        cv2.imwrite(train_ycbcr+folder+'/'+file_,ycbcr)
        # print(test_ycbcr+folder+'/'+file_)
        print('matrix array train image',ycbcr)

fig = plt.figure()
a = fig.add_subplot(1, 2, 1)
imgplot = plt.imshow(img)
a.set_title('Before')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
```

```

a = fig.add_subplot(1, 2, 2)
imgplot = plt.imshow(ybcr)
a.set_title('After')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')

```

Converting to CMYK Color Space

```

os.getcwd()
os.chdir(directory)
!pwd

from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

#####source data#####
test = rgb+'/test/'
train = rgb+'/train/'
test_image = rgb+'/test_image/'

#####destination data#####
test_cmy = '/content/drive/My Drive/datasets/cmy/test'
train_cmy = '/content/drive/My Drive/datasets/cmy/train/'
test_image_cmy = '/content/drive/My Drive/datasets/cmy/test_image/'

def rgb_to_cmy(image):
    rgb_scale = 255
    cmyk_scale = 100

    r,g,b = cv2.split(np.float32(image))

    c = 1 - r / rgb_scale
    m = 1 - g / rgb_scale
    y = 1 - b / rgb_scale

    min_cmy = np.minimum(c, m, y)

    c = (c - min_cmy) / (1 - min_cmy)
    m = (m - min_cmy) / (1 - min_cmy)
    y = (y - min_cmy) / (1 - min_cmy)
    k = (min_cmy)

    img = cv2.merge((c,m,y,k))
    return img

for file_test_image in os.listdir(test_image):
    img = mpimg.imread(test_image+file_test_image)
    cmy = rgb_to_cmy(img)
    plt.imshow(test_image_cmy+file_test_image, cmy)

for folder in os.listdir(test):
    for file_ in os.listdir(test+folder):

```

```

img = mpimg.imread(test+'/'+folder+'/'+file_)
cm_y = rgb_to_cm_y(img)
plt.imsave(test_cm_y+'/'+folder+'/'+file_, cm_y)

for folder in os.listdir(train):
    for file_ in os.listdir(train+folder):
        img = mpimg.imread(train+'/'+folder+'/'+file_)
        cm_y = rgb_to_cm_y(img)
        plt.imsave(train_cm_y+'/'+folder+'/'+file_, cm_y)
        print('matrix array train image',cm_y)

fig = plt.figure()
a = fig.add_subplot(1, 2, 1)
imgplot = plt.imshow(img)
a.set_title('Before')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')
a = fig.add_subplot(1, 2, 2)
imgplot = plt.imshow(cm_y)
a.set_title('After')
plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], orientation='horizontal')

```

Import Module for Deep Learning

```

import sys
import os
import keras
import time
import numpy as np
from keras.preprocessing.image import ImageDataGenerator,
load_img, img_to_array
from keras import optimizers
from keras.models import Sequential, load_model
from keras.layers import Dropout, Flatten, Dense, Activation
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras import callbacks

```

Tuning for All Neural Networks

```

epochs = 15 #1 Epoch = 1 Forward pass + 1 Backward pass for ALL train
ing samples.
img_width, img_height = 150, 150 #size image
batch_size = 32 #number of sample you put into neural network
samples_per_epoch = 1000 #number of samples you want to train in each
epoch
validation_steps = 300 #validation_steps = TotalvalidationSamples / V
alidationBatchSize
classes_num = 2 # number of class
#lr = 0.0004

```

Create model CNN 3 Layer

```

model = Sequential()
model.add(Conv2D(32, (3,3), padding ="same", input_shape=(img_width,
img_height, 3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))

```

```

model.add(Conv2D(32, (3,3), padding="same", activation='relu'))
model.add(Dropout(0.5))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(Flatten())
model.add(Dense(32, activation='relu'))
# model.add(Activation("relu"))
model.add(Dropout(0.5))
model.add(Dense(classes_num, activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

model.summary()

```

Create model CNN 7 Layer

```

model = Sequential()
model.add(Conv2D(32, (3,3), padding="same", input_shape=(img_width,
img_height, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3,3), padding="same", activation='relu'))
model.add(Conv2D(128, (3,3), padding="same", activation='relu'))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
# model.add(Activation("relu"))
model.add(Dropout(0.5))
model.add(Dense(classes_num, activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

```

Create model CNN 10 Layer

```

model = Sequential()
model.add(Conv2D(32, (3,3), padding="same", input_shape=(img_width,
img_height, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(Conv2D(64, (3,3), padding="same", activation='relu'))

```

```

model.add(Conv2D(64, (3,3), padding="same", activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2), dim_ordering='tf'))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3,3), padding="same", activation='relu'))
model.add(Conv2D(128, (3,3), padding="same", activation='relu'))
model.add(Conv2D(128, (3,3), padding="same", activation='relu'))
model.add(Conv2D(128, (3,3), padding="same", activation='relu'))
model.add(Flatten())
model.add(Dense(256, activation='relu'))
# model.add(Activation("relu"))
model.add(Dropout(0.5))
model.add(Dense(classes_num, activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

```

Proses Training (RGB/GRAYSCALE/HSV/XYZ/CMYK/YCrCb)

```

train_data_path_colorsapce = '/content/drive/My Drive/datasets/
colorsapce /train/'
validation_data_path_colorsapce = '/content/drive/My Drive/datasets/
colorsapce /test/'

```

```

train_datagen = ImageDataGenerator(
    rescale=1. / 255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

```

```

test_datagen = ImageDataGenerator(
    rescale=1. / 255,)

```

```

train_generator_colorsapce1 = train_datagen.flow_from_directory(
    train_data_path_colorsapce,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical')

```

```

validation_generator_colorsapce1 = test_datagen.flow_from_directory(
    validation_data_path_colorsapce,
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='categorical')

```

```

log_dir = './tf-log/'
tb_cb = callbacks.TensorBoard(log_dir=log_dir, histogram_freq=0)
cbks = [tb_cb]

```

```

history_rgb = model.fit_generator(
    train_generator_colorsapce1,
    samples_per_epoch=samples_per_epoch,
    epochs=epochs,
    validation_data=(validation_generator_colorsapce1),
    callbacks=cbks,
    validation_steps=validation_steps)

```

```

target_dir = './content/drive/My Drive/model/colorspace/'
if not os.path.exists(target_dir):
    !mkdir colorspace
model.save('./content/drive/My Drive/model/rgb/type_model.h5') //dipilih
sesuai banyaknya layer (Model_3L,Model_7L,Model_11L)
model.save_weights('./content/drive/My Drive/model/rgb/weights_model.h
5') //dipilih sesuai banyaknya layer (weights_3L, weights_7L,weights_10L)

fig = plt.figure()
a = fig.add_subplot(1, 2, 1)
plt.plot(history_rgb.history['accuracy'])
plt.plot(history_rgb.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

a = fig.add_subplot(1, 2, 2)
plt.plot(history_rgb.history['loss'])
plt.plot(history_rgb.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

```

Testing

```

import os
import numpy as np
from keras.preprocessing.image import ImageDataGenerator, load_img, i
mg_to_array
from keras.models import Sequential, load_model
import time

model_path = './content/drive/My Drive/model/colorspace/type_model.h5'
// dipilih sesuai banyaknya layer (Model_3L,Model_7L,Model_10L)
model_weights_path = './content/drive/My Drive/model/colorspace
/weights_model.h5' ////dipilih sesuai banyaknya layer (weights_3L,
weights_7L,weights_11L)
test_data_path_colorspace = './content/drive/My Drive/datasets/
colorspace /test_image/'

#Load the pre-trained models
model = load_model(model_path)
model.load_weights(model_weights_path)

def predict(file):
    x = load_img(file, target_size=(img_width,img_height))
    x = img_to_array(x)
    x = np.expand_dims(x, axis=0)

```

```

array = model.predict(x)
result = array[0]
#print(result)
answer = np.argmax(result)
if answer == 1:
    print("Predicted: mobil dengan persentase", result[1] * 100)
elif answer == 0:
    print("Predicted: bukan mobil dengan persentase", result[0] * 100)

return answer

for i, ret in enumerate(os.walk(test_data_path_ colorspaces)):
    for i, filename in enumerate(ret[2]):
        if filename.startswith("."):
            continue

    print(ret[0] + '/' + filename)
    result = predict(ret[0] + '/' + filename)
    print(" ")

```