LAPORAN AKHIR PENELITIAN
DANA MANDIRI

Peningkatan Resolusi Citra Digital Menggunakan *Deep Neural Network*

Ketua : Dra. Sulistyowati, M.Kom 0324056703
Anggota : Aini Mandasari 1151500019

PROGRAM STUDI TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI INDONESIA
2024

HALAMAN PENGESAHAN

JudulPenelitian	: Peningkatan Resolusi Citra Digital Menggunakan *Deep Neural Network*
JenisPenelitian[a)]	: Dana Mandiri
BidangPenelitian[b)]	: Ilmu Komputer
TujuanSosialEkonomi[c)]	: Aplikasi
Peneliti	
a. Nama Lengkap	: Dra. Sulistyowati, M.Kom
b. NIDN	0324056703
c. JabatanFungsional	: Lektor
d. Program Studi	: Teknik Informatika
e. Nomor HP	08176543515
f. AlamatSurel*(e-mail)*	: liliswiyono2403@gmail.com
Anggota Peneliti 1	
a. Nama Lengkap	: Aini Mandasari
b. NIM	1151500019
c. Institusi	: Institut Teknologi Indonesia
a. InstitusiSumberDana[d)]	: Pribadi
b. BiayaPenelitian	: Rp. 10.000.000,

Kota Tangerang Selatan, Pebruari 2024

Mengetahui,
Kaprodi Teknik Informatika

Muhammad Soleh, S Si, M Kom
NIDN : 0302128902

Ketua Peneliti,

Dra. Sulistyowati, M Kom
NIDN : 0324056703

Menyetujui,
Pusat Riset dan Pengabdian Masyarakat (PRPM) ITI
Kepala,

Prof. Dr. Ir. Ratnawati, M.Eng.Sc.,IPM
NIDN : 0301036303

INSTITUT TEKNOLOGI INDONESIA

Jl. Raya Puspiptek, Tangerang Selatan - 15314
(021) 7562757
www.iti.ac.id institutteknologiindonesia @kampusITI Institut Teknologi Indonesia

SURAT TUGAS

No. : 012/ST-PLT/PRPM-PP/ITI/XII/2023

Pertimbangan	:	Bahwa dalam rangka melaksanakan kegiatan Penelitian bagi Dosen Program Studi Informatika Institut Teknologi Indonesia, perlu dikeluarkan surat tugas.
Dasar	:	1. Pembebanan Tugas dosen Program Studi Informatika; 2. Surat Permohonan Tanggal 13 Desember 2023; 3. Kepentingan Institut Teknologi Indonesia.

D I T U G A S K A N

Kepada	:	Dosen Program Studi Informatika – ITI (Terlampir)
Untuk	:	1. Melaksanakan kegiatan Penelitian pada Semester Ganjil Tahun Akademik 2023/2024; 2. Melaporkan hasil tugas kepada Kepala PRPM-ITI; 3. Dilaksanakan dengan penuh rasa tanggung jawab.

Tangerang Selatan, 14 Desember 2023
Pusat Riset dan Pengabdian Masyarakat
Kepala,

PUSAT RISET DAN PENGABDIAN MASYARAKAT

Prof. Dr. Ir. Joelianingsih, M.T., IPM

Tembusan Yth.
1. Wakil Rektor Bid Akademik, Penelitian dan Kemahasiswaan
2. Ka. Biro SDMO
3. Ka. Prodi Informatika
4. Arsip

Lampiran Surat Tugas
No. 012/ST-PLT/PRPM-PP/ITI/XII/2023
Tanggal 13 Desember 2023

DAFTAR PENELITIAN DOSEN PROGRAM STUDI INFORMATIKA SEMESTER GANJIL THN AKADADEMIK: 2023/2024

NO	TOPIK PENELITIAN	BIDANG	SUSUNAN TIM	SUMBER DANA	JUMLAH DANA (Rp)	KETERLIBATAN PRODI/INSTITUSI LAIN	KETERLIBATAN MAHASISWA
1	Evaluasi Terhadap Metode Penerapan Enterprise Architecture (EA) Framework (Studi kasus: Perusahaan Farmasi Global)	Engineering and Technology	Suryo Bramasto, S.T., M.T	Mandiri	10.000.000	undisclosed	DZIKRI MUSTAQIM (NRP: 1152000040)
2	Pengembangan Game Menggunakan Metode Finite State Machine	Engineering and Technology	Melani Indriasari, S.T., M.Kom	Mandiri	11.500.000	Tidak Ada	Erzhanto (NRP: 1151800048)
3	Implementasi Generative Pre-Trained Transformers Sebagai Pantun Generator	Engineering and Technology	Dino Hariatma Putra, S.T., M.Kom	Mandiri	10.000.000	Tidak Ada	Ricky Khairul Faza (NRP: 1151900034)
4	Implementasi Generative Pretrained Transformer 2 (GPT-2) dalam Pembuatan Abstrak Pada Dokumen Jurnal	Engineering and Technology	Dino Hariatma Putra, S.T., M.Kom	Mandiri	11.000.000	Tidak Ada	Luthfiya Rifqi Bahasuan (NRP: 1151900030)
5	Implementasi Generative Pre-Trained Transformers 2 untuk Mengoreksi Kesalahan Penulisan Bahasa Indonesia Pada Dokumen Jurnal	Engineering and Technology	Dino Hariatma Putra, S.T., M.Kom	Mandiri	10.500.000	Tidak Ada	Gagas Wahyu Ilham Noto (NRP: 1151900046)
6	Optimalisasi Jadwal Pembersihan Mesin Pengemas di PT. HF	Engineering and Technology	Ketua: Ir. Muhami, M.Si., IPM Anggota: Dra. Indrati Sukmadi, M.Sc	Mandiri	10.000.000	Tidak Ada	Nursehat Meilasari (NRP: 1321920008)
7	Penentuan Matriks Similaritas Protein Menggunakan Transfer Learning	Engineering and Technology	Ketua: Yustina Sri Suharini, S.T., M.T Anggota: 1. Dra. Endang Ratnawati D, M.Kom 2. Dra. Sulistyowati, M.Kom 3. Muhamad Ramli, S.T., M.Kom	Mandiri	10.000.000	Tidak Ada	Tidak Ada
8	PEMANFAATAN SIG UNTUK PEMETAAN TAMAN TERBUKA HIJAU DI DKI JAKARTA	Engineering and Technology	Ir. Sumiarti Andri, M.Kom	Mandiri	11.500.000	Pemprov DKI Jakarta	1. Bayu Ardi Setiawan (NRP: 1151600032) 2. Adhy Chandra Utomo (NRP: 1151800048)

Pusat Riset dan Pengabdian Masyarakat
Institusi Teknologi Indonesia
Kepada

Prof. Dr. Ir. Joelianingsih, M.T., IPM

KATA PENGANTAR

Dengan memanjatkan puji syukur ke hadirat Tuhan Yang Maha Esa yang telah melimpahkan rahmat dan anugrah-Nya berupa kesempatan, kemampuan, kesehatan sehingga laporan akhir penelitian dengan dana mandiri bagi Dosen di Institut Teknologi Indonesia dapat diselesaikan. Penelitian ini diberi judul: Peningkatan Resolusi Citra Digital Menggunakan *Deep Neural Network*. Diharapkan dengan dibuatnya laporan penelitian ini dapat memberikan manfaat.

Tangerang Selatan, Pebruari 2024

Penulis

ABSTRAK

Deep Learning merupakan bagian dari bidang ilmu *Machine Learning*, memiliki metode yang dikenal sebagai *Deep Neural Network* yang digunakan untuk melatih jaringan saraf buatan agar bisa bekerja layaknya saraf otak manusia dalam kasus mengenali suatu pola dari sekelompok data yang akan dijadikan suatu pengetahuan untuk sistem. *Deep Neural Network* merupakan salah satu ilmu yang utama untuk dapat menciptakan *Artificial Intelligence* yang lebih cerdas. Citra merupakan gambar pada bidang dua dimensi yang mengandung informasi di dalamnya. Apabila citra memiliki kualitas yang rendah, informasi yang dimiliki citra tersebut akan berkurang bahkan hilang. Khususnya jika citra memiliki kualitas resolusi rendah, untuk meningkatkan kualitas resolusi pada citra dapat digunakan interpolasi *Bilinear upsampling*. Namun citra yang dihasilkan tidak baik. Maka dilakukan proses pengimplementasian model *Low-Light Convolutional Neural Network* untuk memperbaiki kualitas citra hasil proses interpolasi *Bilinear upsampling*.

Kata kunci: *Deep Neural Network, Low-Light Convolutional Neural Network, Machine Learning, Resolusi Citra*

BAB I

PENDAHULUAN

1.1 Latar Belakang

Saat ini, teknologi berkembang dengan sangat cepat. Salah satunya adalah Kecerdasan Buatan yang digunakan untuk membuat sistem cerdas. Bagian dari kecerdasan buatan yang banyak diperbincangkan saat ini adalah *Machine Learning*, dimana *machine learning* adalah suatu area dalam *artificial intelligence* atau kecerdasan buatan yang berhubungan dengan pengembangan teknik-teknik yang bisa diprogramkan dan belajar dari data masa lalu. Penerapan algoritma *machine learning*, dapat diterapkan untuk pengolahan citra. Citra merupakan gambar pada bidang dua dimensi, citra memiliki informasi di dalamnya. Namun pada citra yang dimiliki seringkali mengalami penurunan mutu. Seperti terdapat cacat, kabur (*blur*), kurang tajam, kualitas resolusi yang rendah, dan lain sebagainya. Untuk dapat memiliki citra dengan kualitas resolusi yang tinggi, dibutuhkan kamera yang memiliki spek tinggi. Perkembangan teknologi kamera saat ini memudahkan manusia untuk memillikinya, namun biaya yang harus dikeluarkan cukup tinggi. Bagaimana jika spek kamera yang dimiliki hanya dapat menghasilkan citra dengan kualitas resolusi rendah. Jika citra memiliki resolusi rendah, maka informasi yang dimiliki citra tersebut dapat berkurang bahkan hilang, sehingga citra tidak fleksibel untuk diolah.

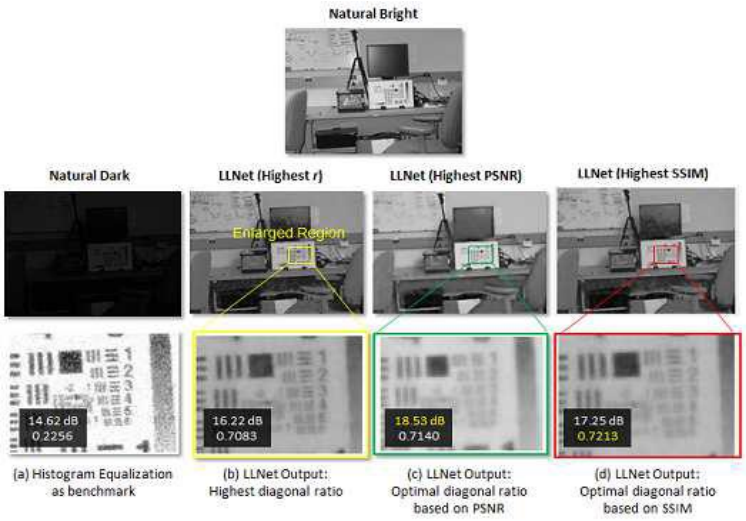
Pengolahan citra atau *image processing* merupakan suatu metode yang digunakan untuk memproses atau memanipulasi gambar dalam bentuk 2 dimensi. Pada proses pengolahan citra, *input* yang dibutuhkan adalah sebuah citra yang akan diproses pada komputer dan menghasilkan *output* berupa citra dengan kualitas lebih

baik daripada citra yang belum diolah. Oleh komputer, citra akan diproses menggunakan operasi yang sesuai dengan tujuan untuk apa citra itu diolah, jika bertujuan untuk transformasi citra, terdapat operasi perbaikan kualitas citra (*image enhancement*) dan operasi pemulihan citra (*image restoration*). Beberapa teknik pengolahan citra yang sudah ada yaitu, teknik *upsampling* digunakan untuk memperbesar citra dengan menambahkan piksel di dalamnya, namun penggunaan teknik ini menghasilkan kualitas ketajaman citra menjadi buruk. Operasi peningkatan kualitas citra (*image enhancement*) bertujuan untuk memperbaiki kualitas citra dengan memanipulasi parameter-parameter pada citra, seperti dalam sebuah kasus perbaikan citra untuk intensitas cahaya yang rendah/gelap menggunakan metode LLCNN (*Low-Light Convolutional Neural Network*) [1] dan metode LL-Net (*Low-Light Net*) [2].

Convolutional networks adalah jaringan saraf tiruan sederhana yang menggunakan proses konvolusi pada perkalian matriks umum untuk setidaknya di satu *layers*-nya [3]. Salah satu penelitian tentang perbaikan citra yang memiliki intensitas cahaya rendah menjadi citra yang terang dan dapat dilihat dengan jelas dilakukan dengan menggunakan metode LLCNN (*Low-Light Convolutional Neural Network*) [1], yang merupakan gabungan dari metode LL-Net dan CNN. Hasil implementasi metode LLCNN dapat dilihat pada gambar 1.1.

Gambar 1.1. Citra LLCNN [1]

Pada kasus yang serupa, sebuah riset dengan metode LL-Net dapat memperbaiki citra gelap menjadi citra yang terang, namun metode ini tidak menerapkan *convolutional layer* dalam prosesnya. Hasil implementasi metode LL-Net dapat dilihat pada gambar 1.2.



Gambar 1.2. Citra hasil LL-Net [2]

1.2 Rumusan Masalah

Dari latar belakang yang telah dijelaskan sebelumnya, dapat dirumuskan beberapa masalah:

1) Apakah citra resolusi rendah dapat ditingkatkan menjadi citra resolusi tinggi.
2) Dapatkah metode *Low-Light Convolutional Neural Network* (LLCNN) mengatasi kelemahan metode *upsampling*.

1.3 Tujuan

Tujuan dari penelitian ini adalah memperbaiki citra yang memiliki kualitas resolusi yang rendah menjadi citra resolusi tinggi dengan menerapkan model *Low-Light Convolutional Neural Network* (LLCNN).

BAB II

LANDASAN TEORI

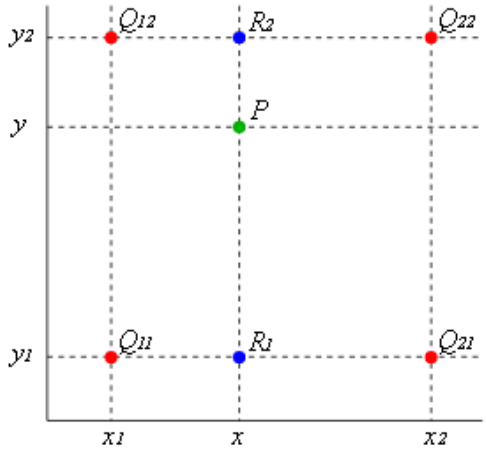
2.1. Interpolasi, *Sampling* atau *Scaling*

Interpolasi merupakan metode untuk merekonstruksi suatu nilai data baru dari nilai data yang telah diketahui dengan fungsi hampiran tertentu. Nilai hasil interpolasi merupakan nilai estimasi. Interpolasi terdiri dari interpolasi *adaptive* dan *non-adaptive*. Interpolasi *adaptive* melakukan proses komputasi secara merata di seluruh pikselnya, sedangkan interpolasi *non-adaptive* melakukan proses komputasi berdasarkan kriteria konten tertentu, misalnya memperlakukan proses yang berbeda pada frekuensi tinggi dan frekuensi rendah pada citra [8].

Salah satu teknik yang digunakan dalam pengolahan citra digital adalah *sampling* atau *scaling*. Saat sebuah citra diperbesar, proses ini dapat disebut sebagai *upsampling.* Proses *upsampling* memerlukan dua langkah, yaitu pembuatan lokasi untuk piksel yang baru dan penerapan *grey levels* ke lokasi baru tersebut. Untuk dapat menerapkan *grey levels* di setiap lokasi piksel yang baru, diambil piksel terdekat dari citra asli lalu perluas ke ukuran yang ditentukan untuk mendapat ukuran citra yang lebih besar. Sedangkan penyusutan citra dapat disebut sebagai proses *downsampling*. Dalam proses *downsampling*, jarak pada grid akan berkurang satu piksel karena ukuran citra menjadi lebih kecil [9].

Salah satu metode untuk menyelesaikan permasalahan dalam menerapkan *grey levels* adalah interpolasi *Bilinear upsampling*. Interpolasi *bilinear upsampling* mengambil nilai terdekat dari nilai-nilai piksel yang diketahui yang mengelilingi

piksel yang tidak diketahui (x, y), seperti pada gambar 2.1. Dibutuhkan rata-rata tertimbang dari 4 piksel ini untuk sampai pada nilai akhir yang diinterpolasi.



Gambar 2.1. *Bilinear upsampling*

Dari ilustrasi pada gambar 2.1 di atas, berikut ini adalah rumus interpolasi *bilinear upsampling*:

- Rumus interpolasi *bilinear upsampling*:

  Interpolasi *linear* pada sumbu x:

$$\boldsymbol{R_1} \quad f(x, y_1) = \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \qquad \textbf{(2.1)}$$

$$\boldsymbol{R_2} \quad f(x, y_2) = \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \qquad \textbf{(2.2)}$$

  Interpolasi *linear* pada sumbu y:

$$\boldsymbol{P} \quad f(x, y) = \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2) \qquad \textbf{(2.3)}$$

$$= \frac{y_2 - y}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})\right)$$

$$+ \frac{y - y_1}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})\right)$$

$$= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \big(f(Q_{11})(x_2 - x)(y_2 - y) + f(Q_{12})(x_2 - x)(y - y_1) + f(Q_{21})(x - x_1)(y_2 - y) + f(Q_{22})(x - x_1)(y - y_1) \big)$$

$$= \frac{1}{(x_2 - x_1)(y_2 - y_1)} [x_2 - x \quad x - x_1] \begin{bmatrix} f(Q_{11}) & f(Q_{12}) \\ f(Q_{21}) & f(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}$$

Keterangan:

- (x, y) = letak koordinat di titik *P*
- $f(x, y)$ = nilai indeks di titik (x,y)

2.2. *Machine Learning* (ML)

Machine Learning adalah bagaimana membuat komputer dapat menyesuaikan tindakannya (seperti membuat prediksi, atau mengendalikan robot) sehingga tindakannya menjadi sangat akurat. Akurasi diukur dengan seberapa baik tindakan yang dipilih dapat mengandung suatu kebenaran. Jika seorang pemain memainkan suatu permainan yang melawan komputer, di permulaan permainan pemain dapat mengalahkan komputer, tetapi setelah melakukan banyak permainan komputer akan mulai dapat mengalahkan pemain tersebut hingga akhirnya pemain ini tidak dapat menang atas komputer. permainan pemain yang menjadi lebih buruk atau komputer yang sedang belajar bagaimana cara memenangkan permainan atas pemain. Setelah komputer belajar mengalahkan pemain, kemampuan ini dapat berkembang lebih baik dan saat bermain dengan pemain lain, komputer akan menggunakan strategi yang sama, sehingga setiap saat bermain dengan pemain baru

kemampuan komputer tidak dimulai kembali dari awal. Tindakan ini dikenal sebagai bentuk generalisasi [10].

Dalam *machine learning* terdapat istilah *Training* dan *Testing*. *Training* (pelatihan) akan dilakukan oleh algoritma *machine learning*. Saat pelatihan, algoritma *machine learning* akan menyesuaikan diri dengan data pelatihan (*train data*) yang diberikan dengan merubah parameternya. *Training* berguna untuk melatih algoritma *machine learning* dengan melakukan pembelajaran dari data secara mandiri agar mengetahui informasi dari data tersebut. Setelah proses *training* dilakukan, proses selanjutnya adalah *testing* (pengujian) yang dilakukan untuk mengevaluasi performa algoritma yang akan diuji menggunakan data pengujian yang berbeda dari data pelatihan sebelumnya.

2.3. *Deep Learning*

Jaringan Syaraf Tiruan atau *Artificial Neural Network* (ANN) merupakan sebuah teknik dalam ML yang sengaja dibuat untuk menyerupai syaraf manusia yang menjadi bagian fundamental dari otak. NN memiliki struktur yang terdiri atas lapis masukan (*input layer*) dan lapis keluaran (*output layer*). Pada setiap lapis terdiri atas satu atau beberapa unit *neuron* yang memiliki sebuah fungsi aktivasi yang berfungsi sebagai "pengaktif *neuron*" untuk menentukan *output* dari unit tersebut. Agar kemampuan dari NN meningkat, lapis tersembunyi (*hidden layer*) dapat ditambah sesuai dengan kebutuhan. NN dapat dilatih dengan menggunakan *data training*. Banyaknya *data training* yang digunakan akan menentukan seberapa bagus kinerja dari NN tersebut. Akan tetapi, kemampuan NN terbatas pada jumlah lapisan, jika jumlah lapisan yang digunakan semakin banyak, maka kapasitas yang

digunakan NN akan semakin tinggi. Banyaknya lapisan yang digunakan akan menimbulkan masalah yaitu, semakin banyaknya jumlah iterasi atau *training* yang dibutuhkan [11].

Teknik *Deep Learning* (DL) dikembangkan untuk mengatasi kekurangan yang ada pada ANN, *deep learning* merupakan sebuah teknik dalam *Neural Network* (NN) yang digunakan untuk mempercepat proses pembelajaran pada NN. Dengan adanya *deep learning*, proses *training* dapat menghemat waktu, karena masalah hilangnya gradien pada *backpropagation* akan semakin rendah. Jenis-jenis *Deep Learning* yang ada antara lain *Deep Auto Encoder, Deep Belief Nets, Convolutional Neural Network,* dan lain sebagainya [11]. Dalam penelitian ini akan digunakan *Convolutional Neural Network*.

2.4. *Convolutional Neural Network* (CNN)

Convolutional Neural Network (Le cun, 1989) juga dikenal sebagai CNN, merupakan jenis jaringan saraf tiruan yang khusus untuk memproses data seperti grid topologi. Contohnya adalah data *time-series* yang dianggap sebagai grid 1 dimensi, dan data citra yang dianggap sebagai grid 2 dimensi. Nama "CNN" menunjukkan bahwa *network* disusun menggunakan operasi matematika yang disebut konvolusi. Konvolusi adalah jenis operasi linear khusus. *Convolutional networks* adalah jaringan saraf tiruan sederhana yang menggunakan proses konvolusi pada perkalian matriks umum untuk setidaknya di satu *layers*-nya [3]. Salah satu tujuan operasi konvolusi adalah untuk mengekstraksi fitur pada citra.

Dalam pengolahan citra, operasi konvolusi merupakan operasi yang mendasar. Konvolusi adalah operasi pada 2 fungsi *f(x)* dan *g(x)* yang dinotasikan sebagai berikut:

$$h(x) = f(x) * g(x) = \int_{-\infty}^{\infty} f(a)g(x-a)da \quad \textbf{(2.4)}$$

Pada fungsi diskrit, notasi konvolusinya adalah sebagai berikut:

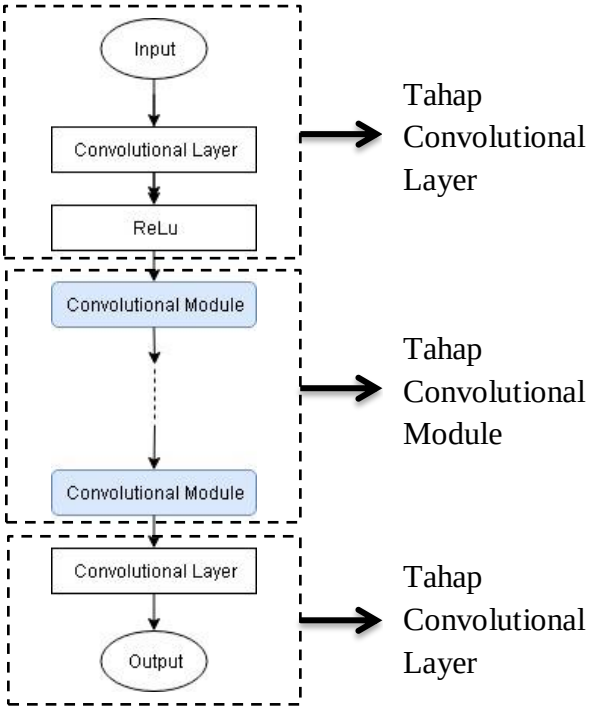
$$h(x) = f(x) * g(x) = \sum_{a=-\infty}^{\infty} f(a)g(x-a) \quad \textbf{(2.5)}$$

Keterangan:

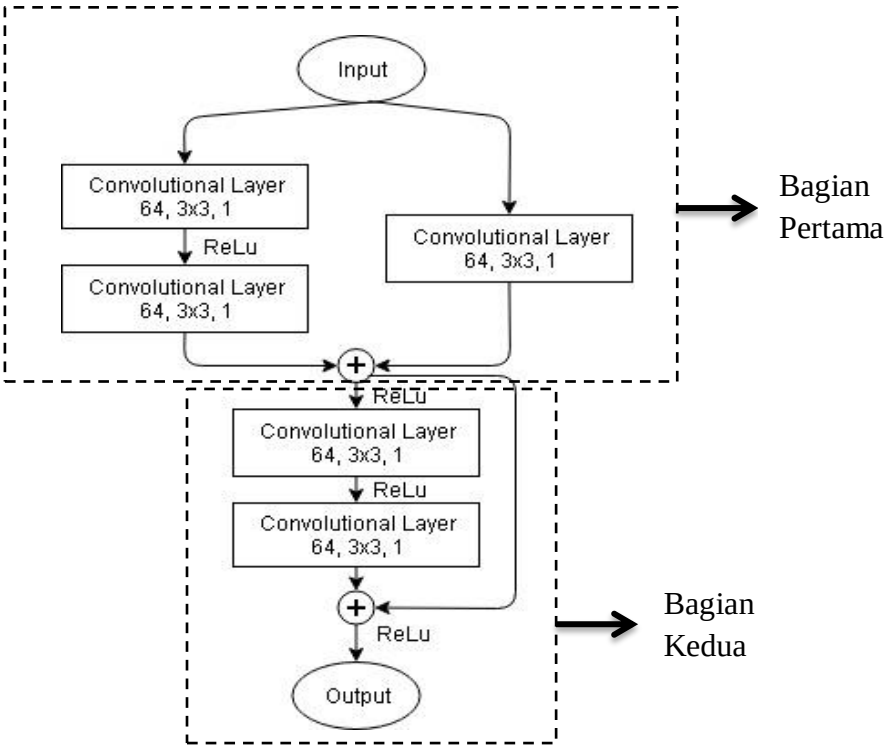
- $*$ = operator konvolusi
- a = *dummy variable*
- $g(x)$ = kernel konvolusi atau kernel penapis (*filter*)
- $f(x)$ = sinyal masukan
- $h(x)$ = keluaran (*output*)

2.5. LLCNN

Tidak semua citra memiliki kualitas yang baik, beberapa diantaranya memiliki tingkat cahaya yang berbeda-beda. Ketika sebuah citra diambil dalam kondisi cahaya yang tidak memadai seperti cahaya yang rendah (*low-light*), nilai-nilai piksel berada dalam rentang dinamis rendah. Hal ini dapat menyebabkan penurunan kualitas. Karena seluruh citra terlihat sangat gelap, sangat sulit untuk mengidentifikasi sebuah objek secara benar [1]. Citra yang demikian itu perlu ditingkatkan mutunya. Salah satu metode yang dapat diterapkan adalah LLCNN. Gambar 2.2 merupakan tahapan dari metode LLCNN dan gambar 2.3 merupakan tahapan dari *convolutional module*.



Gambar 2.2. Arsitektur LLCNN [1]



Gambar 2.3. *Convolutional Module* [1]

BAB III

ANALISIS, PERANCANGAN DAN IMPLEMENTASI LLCNN

3.1. Analisis Masalah

Saat sebuah citra resolusi rendah akan ditingkatkan menjadi citra resolusi tinggi, kasus ini telah memasuki bidang permasalahan dalam pengolahan citra digital (*resolution enhancement*) yang bertujuan untuk memperbaiki kualitas resolusi citra dari citra resolusi rendah menjadi citra resolusi tinggi. Teknik pengolahan citra yang diambil untuk memecahkan kasus ini yaitu *upsampling. Upsampling* digunakan untuk memperbesar citra, karena akan menambah jumlah piksel yang ada di dalam citra dan memperbesar ukuran citra. Namun hasil citra dari proses *upsampling* memiliki kualitas ketajaman yang buruk sehingga citra terlihat pecah. Masalah yang muncul adalah bagaimana cara mengatasi kelemahan yang didapat dari citra hasil proses *upsampling*. Persoalan ini diselesaikan dengan mengimplementasikan model dari agoritma LLCNN.

3.2. Perancangan

3.2.1. Data *Input* dan Data *Output*

Data *input* yang akan digunakan yaitu citra jenis RGB yang memiliki resolusi rendah berukuran ≤ 170x170 piksel, dengan format .jpeg dan .png. data ini dibagi menjadi dua kelompok, yaitu data *training* dan data *testing*. *Data training* yang terdiri dari citra resolusi rendah dan citra target (resolusi tinggi) masing-masing berjumlah 800 citra dan *data testing* berjumlah 200 citra. Data *output* berupa citra RGB berukuran 680x680 piksel atau 4x lebih besar dari ukuran data *input*.

3.2.2. Pre-Processing

Tahap *pre-processing* hanya berlaku untuk citra resolusi rendah yang memiliki ukuran < 170x170 piksel. Pada tahap *pre-processing*, proses *auto padding* akan diberlakukan untuk menambahkan *padding* bernilai 0 pada sisi citra yang dianggap memiliki ukuran < 170x170 piksel, *padding* akan mengisi sisi tersebut hingga ukuran citra menjadi = 170x170 piksel. Tujuan diterapkannya proses *auto padding* yaitu agar matriks *input* dan matriks *output* memiliki ukuran matriks yang sama. Setelah proses *auto padding* dilakukan, selanjutnya citra akan diproses oleh sistem JST.

3.3. Perancangan Arsitektur Jaringan Syaraf Tiruan

Tahap selanjutnya adalah membuat rancangan pada sistem JST yang akan dibangun. Berikut ini adalah rancangan arsitektur JST yang digunakan dalam penelitian ini:

1. Perancangan *Input Layer*

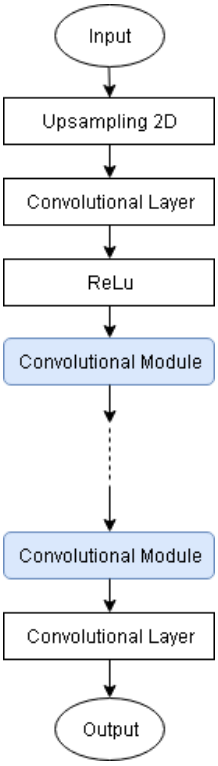
   Sistem JST ini diatur agar menerima *input* berupa citra RGB berukuran 170x170 piksel yang akan diperbesar 4x dari ukuran citra asli menggunakan interpolasi *bilinear upsampling*.

2. Perancangan *Hidden Layer*

   Hidden layer yang terdapat pada sistem ini merupakan sebuah model JST yang diadopsi dari model penelitian LLCNN [1].

3. Perancangan *Output Layer*

   Output layer berupa citra berukuran 680x680. Proses keseluruhan sistem ditampilkan pada gambar 3.1 berikut ini:



Gambar 3.1. Model LLCNN [1]

3.3.1. Proses Pada *Input Layer*

Data *input* berupa citra RGB resolusi rendah berukuran 170x170 piksel akan diperbesar dengan interpolasi *bilinear* 4x dari ukuran citra asli. Misal data citra pada tabel 3.1 berikut ini merupakan contoh perhitungan interpolasi dari citra untuk matriks berukuran 5x5 diubah menjadi 10x10 dengan metode *bilinear upsampling*.

Tabel 3.1. Citra dalam matriks 5x5

	0	1	2	3	4
0	3	1	2	4	1
1	6	7	10	9	2
2	10	10	10	0	7
3	7	2	5	9	5
4	10	4	2	2	0

1) Membuat matriks 10x10 yang akan menjadi citra hasil. Lalu nilai yang terletak paling ujung atas kanan, atas kiri, bawah kanan, dan bawah kiri

pada matriks sebelumnya, diletakkan di sisi matriks tujuan seperti pada tabel 3.2.

Tabel 3.2. Citra dalam matriks 5x5 diperbesar x2 menjadi matriks 10x10

	0	1	2	3	4	5	6	7	8	9
0	**3**									**1**
1										
2										
3										
4										
5										
6										
7										
8										
9	**10**									**0**

2) Untuk meletakkan nilai-nilai yang ada pada matriks sebelumnya ke matriks 10x10, maka terlebih dahulu ditentukan dimana letak indeksnya. Berikut ini cara mencari letak indeks yang baru:

$$indeks_{baru} = \frac{indeks_0 + indeks_n}{2} \quad \textbf{(3.1)}$$

$$= \frac{0+9}{2} = 4{,}5$$

Keterangan :

- $n = 10 - 1$

Didapatlah indeks kolom dan baris dinilai 4,5. Maka letakkan nilai matriks sebelumnya yang bersesuaian dengan letak indeks tersebut sehingga diperolah hasil seperti pada tabel 3.3.

Tabel 3.3. Citra dalam matriks dengan indeks baru

	0	1	2	3	4	4,5	5	6	7	8	9
0	3					2					1
1											
2											
3											
4											
4,5	10					10					7
5											
6											
7											
8											
9	10					2					0

3) Lakukan seperti langkah sebelumnya untuk mencari letak indeks baru di bagian yang lain. Indeks baru ini berfungsi sebagai indeks pembantu untuk mencari nilai-nilai baru. Setelah nilai baru didapat, maka indeks pembantu akan dihapus. Hasil dari tahap ini ditampilkan pada tabel 3.4.

Tabel 3.4. Citra dalam matriks dengan seluruh lokasi untuk indeks pembantu

	0	1	2	2,25	3	4	4,5	5	6	6,75	7	8	9
0	3			1			2			4			1
1													
2													
2,25	6			7			10			9			2
3													
4													
4,5	10			10			10			0			7
5													
6													
6,75	7			2			5			9			5
7													
8													
9	10			4			2			2			0

4) Berikut ini cara mencari nilai-nilai yang belum diketahui diantara nilai-nilai yang sudah diketahui. Yaitu dengan menngunakan rumus interpolasi dari (2.1) hingga (2.3).

$$nilai\ di\ (0{,}5) = \frac{6{,}75 - 5}{6{,}75 - 4{,}5}.2 + \frac{5 - 4{,}5}{6{,}75 - 4{,}5}.4 = 2{,}44$$

$$nilai\ di\ (0{,}6) = \frac{6{,}75-6}{6{,}75-4{,}5}.2 + \frac{6-4{,}5}{6{,}75-4{,}5}.4 = 3{,}33$$

Setelah dilakukan untuk seluruh nilai yang belum diketahu, akan diperoleh data citra seperti pada tabel 3.5 berikut ini merupakan tabel citra dalam matriks berukuran 10x10 yang telah diinterpolasi, matriks ini terdiri dari nilai piksel yang baru dan nilai piksel pembantu.

Tabel 3.5. Citra dalam matriks yang telah diinterpolasi

RESIZING MATRIX(5X5 -> 10X10) = BILINEAR INTERPOLATION													
	0	1	2	2,25	3	4	4,5	5	6	6,75	7	8	9
0	**3**	2,11	1,22	**1**	1,33	1,78	**2**	2,44	3,33	**4**	3,67	2,33	**1**
1	4,33	4,04	3,74	3,67	4,3	5,14	5,56	5,7	6	6,22	5,69	3,57	1,44
2	5,67	5,96	6,26	6,33	7,26	8,49	9,11	8,96	8,67	8,44	7,72	4,8	1,89
2,25	**6**	6,44	6,89	**7**	8	9,33	**10**	9,78	9,33	**9**	8,22	5,11	**2**
3	7,33	7,63	7,93	8	8,67	9,56	10	9,11	7,33	6	5,74	4,7	3,67
4	9,11	9,21	9,31	9,33	9,56	9,85	10	8,22	4,67	2	2,43	4,16	5,89
4,5	**10**	10	10	**10**	10	10	**10**	7,78	3,33	**0**	0,78	3,89	**7**
5	9,33	8,84	8,35	8,22	8,44	8,74	8,89	7,36	4,3	2	2,51	4,53	6,56
6	8	6,52	5,04	4,67	5,33	6,22	6,67	6,52	6,22	6	5,96	5,81	5,67
6,75	**7**	4,78	2,56	**2**	3	4,33	**5**	5,89	7,67	**9**	8,56	6,78	**5**
7	7,33	5,06	2,79	2,22	3,04	4,12	4,67	5,46	7,04	8,22	7,8	6,12	4,44
8	8,67	6,2	3,73	3,11	3,19	3,28	3,33	3,73	4,52	5,11	4,79	3,51	2,22
9	**10**	7,33	4,67	**4**	3,33	2,44	**2**	2	2	**2**	1,78	0,89	**0**

5) Setelah seluruh nilai baru didapatkan, maka seluruh indeks pembantu beserta nilai pikselnya dihapus. Tabel 3.6 berikut ini merupakan hasil akhir proses interploasi *bilinear.*

Tabel 3.6. Citra dalam matriks 10x10

RESIZED MATRIX										
	0	1	2	3	4	5	6	7	8	9
0	**3**	2,11	1,22	1,33	1,78	2,44	3,33	3,67	2,33	**1**
1	4,33	4,04	3,74	4,3	5,14	5,7	6	5,69	3,57	1,44
2	5,67	5,96	6,26	7,26	8,49	8,96	8,67	7,72	4,8	1,89
3	7,33	7,63	7,93	8,67	9,56	9,11	7,33	5,74	4,7	3,67
4	9,11	9,21	9,31	9,56	9,85	8,22	4,67	2,43	4,16	5,89
5	9,33	8,84	8,35	8,44	8,74	7,36	4,3	2,51	4,53	6,56
6	8	6,52	5,04	5,33	6,22	6,52	6,22	5,96	5,81	5,67
7	7,33	5,06	2,79	3,04	4,12	5,46	7,04	7,8	6,12	4,44
8	8,67	6,2	3,73	3,19	3,28	3,73	4,52	4,79	3,51	2,22
9	**10**	7,33	4,67	3,33	2,44	2	2	1,78	0,89	**0**

3.3.2. Proses Pada *Hidden Layer*

1) Setelah citra hasil interpolasi dilakukan untuk masing-masing *channel* R,G dan B. Masing-masing *channel* ini di konvolusi dengan menggunakan 64 *filters* berukuran 3x3 dan diperoleh 64 matriks untuk setiap *channel* R,G dan B.
2) Menjumlahkan *channel* R,G dan B untuk setiap *filter*. Diperoleh matriks RGB berjumlah 64.
3) Menerapkan fungsi aktivasi ReLU untuk seluruh 64 matriks RGB.
4) Pada *convolutional module layer* bagian pertama, setiap matriks diterapkan 2 proses:
    a. Proses pertama adalah melakukan konvolusi dengan 64 *filters* berukuran 3x3. Hasil dari proses konvolusi untuk setiap *filter* akan diperoleh matriks sejumlah 64 yang akan di aktivasi dengan ReLU. Setelahnya kembali diproses dengan konvolusi yang sama seperti sebelumnya.
    b. Proses kedua melakukan konvolusi yang sama namun hanya satu kali tanpa proses aktivasi.

Kemudian hasil dari kedua proses ini (4a dan 4b) dijumlahkan dan diperoleh 64 matriks.

5) Pada *convolutional module layer* bagian kedua, setiap matriks diterapkan 2 proses:
   a. 64 matriks sebelumnya akan di aktivasi dengan ReLU. Kemudian dikonvolusi dengan menngunakan 64 *filters* berukuran 3x3. Proses ini dilakukan sebanyak 2 kali dengan proses aktivasi.
   b. 64 matriks pada bagian ini tidak diterapkan proses apapun.

   Hasil dari kedua proses ini (5a dan 5b) dijumlahkan dan diperoleh 64 matriks. Selanjutnya 64 matriks ini diaktivasi dengan ReLU.
6) Ke-64 matriks hasil proses sebelumnya dikonvolusi dengan 3 *filters* berukuran 3x3 untuk masing-masing *filters*. Hasil dari masing-masing *filters* dijumlahkan sehingga diperoleh 3 matriks sebagai citra R, G dan B.

3.3.3. Proses Pada *Output Layer*

Setelah proses pada *hidden layer* selesai, dilakukan proses yang ada pada *output layer* untuk dapat menghasilkan *output* dari sistem JST. Berikut ini adalah contoh perhitungan operasi konvolusi dengan menggunakan 6 *filters* (contoh perhitungan ini merupakan nilai yang baru, bukan hasil proses interpolasi dari tabel 3.6 sebelumnya) Tabel 3.7, 3.9 dan 3.11 merupakan matriks dari citra RGB berukuran 10x10 yang telah diberi *padding* bernilai 0 disekelilingnya, sehingga menjadi matriks berukuran 12x12. Tabel 3.8, 3.10 dan 3.12 merupakan *filter*

sebanyak 6 *filter* berukuran 3x3 untuk masing-masing matriks R,G dan B. Tabel 3.13 merupakan bias berukuran 1x1 sebanyak 6.

Tabel 3.7. Citra dalam matriks R

R												
	0	0	0	0	0	0	0	0	0	0	0	0
	0	0,424	0,041	0,665	0,756	0,445	0,828	0,122	0,437	0,327	0,862	0
	0	0,833	0,076	0,741	0,841	0,209	0,898	0,168	0,949	0,23	0,082	0
	0	0,422	0,367	0,062	0,724	0,863	0,567	0,038	0,744	0,949	0,732	0
	0	0,749	0,497	0,6	0,015	0,252	0,717	0,505	0,675	0,651	0,459	0
	0	0,691	0,926	0,055	0,195	0,46	0,493	0,214	0,2	0,399	0,939	0
	0	0,377	0,707	0,815	0,137	0,164	0,986	0,206	0,401	0,277	0,487	0
	0	0,033	0,254	0,62	0,246	0,186	0,446	0,81	0,189	0,804	0,391	0
	0	0,634	0,902	0,977	0,53	0,568	0,87	0,378	0,343	0,02	0,581	0
	0	0,271	0,645	0,809	0,119	0,765	0,108	0,883	0,991	0,36	0,824	0
	0	0,439	0,544	0,989	0,131	0,588	0,06	0,231	0,422	0,958	0,244	0
	0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.8. *Filters* matriks R

Filter 0		
1	-1	0
0	1	-1
1	-1	0

filter 1		
1	0	0
0	1	-1
0	-1	-1

filter 2		
1	1	1
0	1	1
1	1	-1

filter 3		
0	0	-1
0	0	-1
-1	0	-1

filter 4		
1	-1	-1
0	1	0
0	-1	1

filter 5		
-1	1	-1
1	-1	-1
1	0	0

Tabel 3.9. Citra dalam matriks G

G												
	0	0	0	0	0	0	0	0	0	0	0	0
	0	0,067	0,273	0,345	0,434	0,61	0,047	0,29	0,568	0,554	0,99	0
	0	0,05	0,036	0,567	0,772	0,705	0,751	0,892	0,011	0,405	0,657	0
	0	0,767	0,541	0,233	0,183	0,416	0,744	0,529	0,463	0,54	0,123	0
	0	0,645	0,768	0,55	0,192	0,013	0,633	0,73	0,777	0,988	0,324	0
	0	0,368	0,508	0,133	0,769	0,347	0,828	0,158	0,017	0,778	0,806	0
	0	0,431	0,925	0,433	0,554	0,368	0,806	0,515	0,277	0,025	0,624	0
	0	0,32	0,212	0,349	0,099	0,158	0,619	0,548	0,423	0,346	0,479	0
	0	0,496	0,059	0,966	0,564	0,739	0,506	0,093	0,104	0,282	0,101	0
	0	0,038	0,764	0,877	0,443	0,739	0,723	0,398	0,607	0,095	0,715	0
	0	0,645	0,114	0,01	0,66	0,348	0,706	0,34	0,928	0,451	0,572	0
	0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.10. *Filters* matriks G

Filter 0		
-1	1	0
1	0	-1
-1	1	0

filter 1		
1	-1	-1
-1	0	-1
-1	0	0

filter 2		
-1	-1	-1
-1	0	0
-1	-1	1

filter 3		
1	0	-1
1	0	0
1	-1	1

filter 4		
0	1	0
-1	0	-1
-1	0	0

filter 5		
-1	-1	0
0	1	-1
-1	-1	0

Tabel 3.11. Citra dalam matriks B

B												
	0	0	0	0	0	0	0	0	0	0	0	0
	0	0,053	0,316	0,137	0,826	0,284	0,412	0,995	0,831	0,013	0,945	0
	0	0,024	0,733	0,669	0,383	0,039	0,447	0,383	0,228	0,836	0,308	0
	0	0,194	0,051	0,03	0,976	0,565	0,829	0,348	0,98	0,972	0,087	0
	0	0,854	0,559	0,136	0,332	0,107	0,041	0,727	0,52	0,882	0,698	0
	0	0,684	0,289	0,971	0,852	0,331	0,692	0,165	0,453	0,965	0,286	0
	0	0,511	0,317	0,766	0,826	0,686	0,81	0,771	0,349	0,991	0,199	0
	0	0,229	0,222	0,707	0,459	0,26	0,101	0,383	0,926	0,694	0,46	0
	0	0,519	0,533	0,146	0,535	0,071	0,244	0,49	0,174	0,146	0,773	0
	0	0,725	0,478	0,916	0,333	0,222	0,985	0,665	0,983	0,309	0,86	0
	0	0,061	0,57	0,201	0,051	0,712	0,995	0,27	0,909	0,345	0,062	0
	0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.12. *Filters* matriks B

Filter 0		
0	-1	-1
-1	1	1
0	-1	-1

filter 1		
0	1	1
0	-1	1
1	-1	-1

filter 2		
0	0	1
0	-1	0
0	-1	0

filter 3		
0	0	1
-1	-1	-1
1	1	0

filter 4		
-1	0	1
1	-1	0
-1	-1	1

filter 5		
0	0	-1
1	-1	0
-1	0	1

Tabel 3.13. Bias pada *convolutional layer* pertama

bias 0	bias 1	bias 2	bias 3	bias 4	bias 5
0	1	0	-1	0	1

Tabel 3.14 - 3.19 merupakan *output* matriks R hasil proses konvolusi antara tabel 3.7 berisi matriks R dengan tabel 3.8 berisi 6 *filter* untuk matriks R.

Tabel 3.14. Konvolusi pertama - *Output* matriks R *filter* ke-0

Output **matriks R *filter* ke-0**									
-0,45	0,133	-0,76	0,211	0,249	0,017	0,415	-0,67	0,185	1,01
-0,09	-0,23	-0,42	-0,12	-0,52	0,643	0,454	-0,3	0,053	-0,24
-1,53	1,314	-1,43	0,346	0,692	-0,63	0,236	-1,16	0,961	1,071
-0,86	-0,28	1,761	-1,04	-0,87	0,474	0,638	-0,67	-0,21	0,136
-1,36	0,794	-0,35	0,998	-0,3	-1,01	1,005	-0,56	-0,39	0,92
-1,05	-0,56	1,185	0,206	-1,03	0,487	-0,28	0,759	-1,02	0,36
-1,23	-0,96	0,191	1,185	-0,33	-1,49	1,893	-0,77	0,86	-0,38
-0,57	-0,67	-0,08	1,026	-0,89	0,888	-1,1	0,835	-0,54	0,53
-1,45	-0,54	0,171	0,659	0,162	-0,55	0,213	0,475	-0,68	0,977
-0,38	-0,82	0,694	0,233	-0,12	0,486	-0,97	-0,64	1,345	-0,22

Tabel 3.15. Konvolusi pertama - *Output* matriks R *filter* ke-1

Output matriks R *filter* ke-1									
-0,53	-1,44	-1,67	-0,74	-1,49	-0,36	-1,43	-1,07	-0,85	0,78
-0,03	-0,67	-0,85	-0,29	-1,36	0,571	-0,73	-0,85	-1,1	-0,32
-1,19	0,042	-1,2	0,336	0,169	-0,48	-0,99	-1,36	0,057	0,502
-1,36	-0,66	0,703	-0,83	-0,69	0,367	-0,02	-0,54	-0,4	0,469
-1,32	0,099	-0,59	0,033	-1,17	-0,66	0,123	-0,37	-0,63	1,102
-0,62	-0,29	0,739	-0,4	-1,26	-0,02	-0,7	-0,66	-1,21	0,495
-1,76	-1,87	-0,43	-0,22	-1,56	-1,45	0,887	-0,77	0,213	0,087
-1,18	-1,5	-0,23	-0,3	-0,93	-0,31	-1,39	-0,22	-1,55	0,561
-1,36	-1,06	0,473	-0,39	0,538	-0,5	0,108	-0,37	-1,32	0,6
-0,1	-0,17	1,503	0,352	0,647	0,594	-0,08	0,347	1,705	0,604

Tabel 3.16. Konvolusi pertama - *Output* matriks R *filter* ke-2

Output matriks R *filter* ke-2									
1,222	0,874	1,397	2,575	1,425	1,89	0,676	1,651	2,286	1,174
1,429	2,674	2,751	2,84	4,156	3,853	2,365	1,898	2,9	2,952
1,951	2,726	3,526	3,741	2,927	2,342	3,343	3,57	3,809	2,153
1,8	3,51	2,555	1,704	3,283	3,428	3,036	3,072	3,194	3,478
2,533	3,095	2,746	2,308	1,251	3,125	3,103	2,76	3,313	2,813
2,48	2,862	2,755	1,69	2,284	2,182	2,582	1,686	2,905	3,021
1,103	3,333	3,873	2,487	2,147	3,673	3,498	2,578	2,143	1,756
1,45	2,893	3,962	2,314	3,092	2,68	2,166	3,682	2,513	2,96
2,349	3,962	4,739	3,49	3,498	3,223	3,334	1,787	3,264	2,628
1,899	3,258	2,693	2,412	1,64	2,047	2,635	3,614	3,377	1,428

Tabel 3.17. Konvolusi pertama - *Output* matriks R *filter* ke-3

Output matriks R *filter* ke-3									
-0,12	-2,24	-1,67	-1,39	-2,57	-0,5	-2,28	-0,72	-1,89	-0,23
-0,48	-1,89	-2,69	-1,58	-3,02	-1,19	-2,7	-1,54	-2,42	-0,95
-0,94	-2,15	-2,08	-1,92	-2,2	-0,96	-3,09	-2,33	-1,95	-0,65
-1,79	-1,41	-1,86	-1,63	-1,97	-1,22	-2,11	-2,21	-2,33	-0,4
-2,13	-1,85	-1,05	-1,69	-2,33	-1,09	-2,26	-1,53	-2,29	-0,28
-1,89	-1,52	-0,83	-1,43	-2,17	-1,42	-1,24	-2,29	-2,01	-0,8
-1,86	-3,05	-1,81	-1,89	-2,83	-1,96	-1,8	-1,48	-1,8	-0,02
-1,8	-2,68	-1,54	-2,33	-1,54	-2,84	-1,63	-2,07	-2,79	-0,36
-2,09	-3,21	-1,32	-2,91	-1,17	-2,08	-1,82	-1,57	-2,07	-0,96
-1,19	-1,8	-0,25	-1,35	-0,17	-1,11	-1,41	-1,32	-1,07	0

Tabel 3.18. Konvolusi pertama - *Output* matriks R *filter* ke-4

Output matriks R *filter* ke-4									
-0,33	0,706	0,765	0,124	1,134	0,098	0,904	-0,28	0,179	0,78
0,314	-0,51	0,022	0,444	-0,6	-0,14	1,143	0,512	-0,74	-1,18
-0,74	0,486	-2,03	0,652	1,062	-0,5	-0,01	-0,29	1,395	0,42
0,195	-0,38	0,321	-1,24	-0,42	0,696	0,276	-0,78	0,255	-0,26
-0,22	0,687	-0,74	0,555	0,328	-1,26	-0,06	-0,75	0,175	0,643
-1,02	0,782	1,118	-0,52	-0,33	1,103	-0,34	0,631	-1,27	-0,44
-0,78	-0,82	-0,07	0,798	-0,53	-1,07	1,155	-0,6	1,001	-0,4
0,721	0,226	-0,33	1,363	-0,48	0,574	-0,07	-0,47	-0,52	0,17
-1,16	-0,15	-0,65	0,455	-0,67	-0,4	1,223	1,542	-0,61	0,019
-0,48	-0,64	0,705	0,056	-0,17	-0,17	-1,53	-0,05	0,766	-0,22

Tabel 3.19. Konvolusi pertama - *Output* matriks R *filter* ke-5

Output matriks R *filter* ke-5									
-0,46	0,552	-1,3	0,205	0,324	-0,3	1,167	-0,47	0,197	-0,3
-0,53	-0,61	-1,27	-0,6	-0,68	0,267	-0,8	-0,99	0,41	1,631
-0,03	-0,76	-0,1	-1,03	-2,22	1,032	-1,18	-0,6	-1,06	0,72
-1,19	0,227	-0,22	0,187	-1,19	-0,84	-1,24	-0,85	-0,76	0,373
-1,36	-0,77	1,472	-0,62	-1,1	-0,12	0,178	-0,66	-1,22	-0,46
-1,32	-0,93	-1,06	0,813	-1	-1,02	0,346	-0,07	-0,92	1,134
-0,62	-0,69	0,262	0,321	-0,81	0,113	-0,86	0,113	-1,27	0,645
-1,76	-1,37	0,16	0,128	-1,29	-0,47	0,432	-0,53	0,957	-0,61
-1,18	-1,45	-0,2	-0,1	-1,45	0,286	-2,54	-0,29	-0,67	1,055
-1,36	-1,53	-0,53	-1,18	0,02	-1,24	-0,81	-1,4	-2,24	1,178

Tabel 3.20 - 3.25 merupakan *output* matriks G hasil proses konvolusi antara tabel 3.9 berisi matriks G dengan tabel 3.10 berisi 6 *filter* untuk matriks G.

Tabel 3.20. Konvolusi pertama - *Output* matriks G *filter* ke-0

Output matriks G *filter* ke-0									
-0,22	-0,29	0,37	-0,06	0,321	0,365	-0,38	-1,14	-0,03	0,806
0,799	-0,54	-0,97	-0,1	0,429	-0,42	0,767	0,699	-0,58	0,425
0,154	0,643	0,671	-0,34	-0,81	0,552	0,519	-0,84	0,944	0,129
0,367	0,009	-0,11	1,123	-0,63	0,093	-1,03	-0,46	1,291	0,599
0,568	0,851	-0,97	-0,45	-0,42	1,247	0,617	-0,81	-0,83	0,714
-0,24	0,03	0,132	0,452	-0,62	0,795	-0,21	0,225	0,338	0,187
0,716	0,027	0,529	-0,09	-0,53	-0,19	-0,51	-0,02	-0,13	0,764
0,3	0,149	-0,26	-0,46	0,412	1,091	0,007	-0,1	-0,59	1,036
0,377	-1,81	1,124	0,386	-0,42	0,466	-0,66	0,903	-0,41	0,035
-0,08	1,361	-0,43	-0,77	0,25	-0,01	-0,55	0,098	-0,16	1,072

Tabel 3.21. Konvolusi pertama - *Output* matriks G *filter* ke-1

Output matriks G *filter* ke-1									
-0,27	-0,46	-0,74	-1,52	-1,25	-1,61	-1,37	-1,74	-1,57	-0,96
-0,38	-1,93	-1,85	-2,21	-1,93	-1,74	-2,32	-2,66	-2,11	-1,38
-0,63	-2,2	-2,79	-2,11	-1,8	-1,9	-1,99	-1,32	-2,42	-1,78
-2,08	-1,57	-1,34	-1,06	-2,57	-1,95	-2,49	-2,35	-1,32	-1,35
-1,92	-1,61	-2,18	-0,57	-2,6	-2,22	-2,52	-2,49	-1,64	-0,14
-1,8	-1,46	-2,08	-2,13	-1,86	-1,68	-1,05	-1,73	-2,89	-0,4
-1,57	-2,09	-0,43	-1,96	-1,9	-2,4	-1,53	-0,77	-1,38	-1,23
-0,59	-1,74	-1,62	-2,49	-2,19	-2,58	-1,69	-0,99	-1,21	-0,51
-1,32	-2,09	-2,79	-1,96	-2,51	-1,35	-1,73	-1,13	-2,53	-0,37
-0,92	-2,26	-1,33	-0,66	-2,39	-1,07	-1,92	-1,1	-1,7	-1,07

Tabel 3.22. Konvolusi pertama - *Output* matriks G *filter* ke-2

Output matriks G *filter* ke-2									
-0,01	0,414	-0,1	-0,98	-1,16	-1,17	-1,68	-0,79	-0,33	-1,62
-0,57	-1,81	-1,68	-1,96	-1,72	-2,28	-2,47	-2,76	-3	-2,61
0,037	-2,28	-3,04	-3,01	-1,98	-2,68	-2,98	-2,36	-2,98	-2,91
-1,17	-2,93	-1,6	-1,94	-1,82	-2,72	-3,34	-1,66	-1,89	-3,24
-0,92	-3,25	-2,82	-1,51	-1,72	-2,38	-4,01	-3,42	-1,78	-2,74
-0,98	-1,62	-2,8	-1,97	-2,13	-1,93	-2,55	-2,09	-2,17	-2,43
-1,79	-1,7	-2,58	-2,49	-2,62	-3	-2,71	-1,28	-1,63	-1,38
0,193	-1,3	-1,92	-2,15	-1,9	-3,13	-2,61	-2,32	-1,34	-1,92
-1,09	-2,31	-1,82	-3,47	-2,55	-2,79	-1,54	-1,69	-1,9	-1,5
-0,8	-2,33	-2,2	-2,07	-2,57	-2,21	-2,44	-1,44	-2,35	-1,26

Tabel 3.23. Konvolusi pertama - *Output* matriks G *filter* ke-3

Output matriks G *filter* ke-3									
-0,01	0,649	0,513	0,846	1,251	1,457	-0,08	1,576	0,831	0,302
-0,5	0,232	0,365	0,768	1,67	1,226	0,908	1,234	-0,36	1,376
0,087	0,677	0,215	0,466	1,016	0,34	2,164	1,956	-0,07	1,609
-0,4	1,172	2,269	0,078	0,881	-0,42	1,6	1,638	1,162	1,5
-0,27	0,403	2,129	0,917	1,321	-0,29	1,251	0,162	1,346	1,167
-0,62	1,123	0,627	0,627	1,056	0,644	2,111	0,365	0,045	0,669
-1,36	1,722	0,24	1,555	0,178	0,335	1,665	1,31	9E-04	0,551
0,514	0,619	0,502	2,33	0,472	0,762	1,634	0,181	1,276	0,007
-0,59	0,111	1,024	0,802	1,519	1,367	2,42	0,073	1,659	0,256
-0,76	-0,19	0,436	0,148	0,38	0,689	0,822	0,644	0,821	0,546

Tabel 3.24. Konvolusi pertama - *Output* matriks G *filter* ke-4

Output matriks G *filter* ke-4									
-0,27	-0,46	-0,74	-1,52	-1,25	-1,61	-1,37	-1,74	-1,57	-0,96
0,032	-1,11	-1	-1,07	-1,1	-1,97	-1,22	-1,26	-0,58	0,045
-0,49	-1,61	-0,92	-0,43	-0,41	-0,21	-0,95	-1,79	-0,96	-0,87
-0	-1,02	-1,23	-0,51	-1,18	-0,35	-1,71	-1,41	-0,58	-1,64
0,137	-0,16	-1,65	-0,72	-2,14	-0,24	-0,92	-0,67	-0,11	-0,48
-0,56	-0,68	-1,56	-0,38	-1,11	-0,21	-1,54	-1,07	-0,55	0,436
0,219	-0,24	0,063	-0,92	-0,91	-0,64	-1,03	-0,71	-0,98	-0
0,262	-1,29	-1,04	-2,48	-1,36	-0,95	-0,78	-0,35	-0,47	0,103
-0,27	-1,5	-0,36	-1,06	-1,09	-0,98	-1,94	-0,73	-1,97	-0,44
-0,08	0,109	0,102	0,085	-0,63	0,035	-1,24	-0,18	-1,41	0,263

Tabel 3.25. Konvolusi pertama - *Output* matriks G *filter* ke-5

Output matriks G *filter* ke-5									
-0,26	-0,16	-0,69	-1,51	-0,91	-1,7	-1,92	-0,89	-0,85	-0,07
-0,82	-2,18	-1,6	-1,13	-1,69	-1,96	-0,73	-2,24	-2,38	-1,55
-0,47	-1,19	-1,87	-2,31	-2,01	-1,89	-2,94	-2,49	-1,76	-2,25
-1,26	-1,97	-1,06	-1,14	-2,33	-2,43	-2,31	-1,38	-1,14	-1,92
-1,22	-2,39	-3,31	-1,31	-1,61	-1,15	-2,54	-3,06	-2,09	-1,15
-1,18	-0,92	-1,32	-1,16	-1,81	-1,66	-1,91	-0,89	-2,16	-1,79
-0,82	-2,05	-2,13	-2,57	-2,69	-2,35	-1,8	-0,91	-0,82	-0,55
0,079	-2,24	-1,8	-1,94	-1,21	-1,83	-2,3	-2,16	-1,29	-1,53
-1,87	-1,43	-0,71	-2,5	-2,3	-1,97	-1,85	-0,95	-2,39	-0,69
0,493	-0,7	-2,29	-1,01	-1,54	-1,1	-1,71	-0,53	-0,82	-0,24

Tabel 3.26 - 3.31 merupakan *output* matriks B hasil proses konvolusi antara tabel 3.11 berisi matriks B dengan tabel 3.12 berisi 6 *filter* untuk matriks B.

Tabel 3.26. Konvolusi pertama - *Output* matriks B *filter* ke-0

Output matriks B *filter* ke-0									
-0,39	-1	-0,41	0,55	-0,62	0,293	0,803	-1,22	-1,02	0,624
0,143	0,843	-1,65	-2,9	-1,99	-1,79	-2,99	-2,11	-1,1	-1,56
-1,93	-2,21	-0,56	0,649	-0,22	-0,99	-1,36	-0,86	-2,65	-1,89
0,196	-1,5	-2,92	-2,42	-2,6	-1,37	-0,74	-2,7	-1,25	-0,56
-1,27	-1,2	-0,53	-1,74	-1,47	-1,82	-2,44	-1,49	-1,97	-1,58
-0,6	-1,62	-1,71	-1,16	-0,72	-0,45	-1,62	-2,47	-1,56	-1,54
-1,43	-1,06	-1,33	-2,11	-1,91	-2,09	-0,58	-0,42	-1,88	-1,21
-0,6	-2,16	-2,27	-0,81	-1,79	-1,47	-2,54	-3,08	-1,58	-0,69
-0,48	-0,78	-0,16	-1,73	-1,15	-0,57	-1,18	-0,95	-1,14	-0,28
-0,57	-0,68	-1,57	0,007	0,45	-1,1	-1,46	-0,31	-1,67	-1,14

Tabel 3.27. Konvolusi pertama - *Output* matriks B *filter* ke-1

Output matriks B *filter* ke-1									
-0,49	-1,56	0,368	-0,29	0,024	-0,21	-0,33	-1,5	0,016	-0,42
0,832	0,504	-0,28	-0,74	0,688	0,731	1,173	-0,15	0,351	1,522
-0,8	1,541	2,091	-0,29	0,934	-0,31	0,037	0,382	-0,8	0,406
-1,02	-0,92	-0,33	1,104	1,156	1,336	1,194	1,061	0,077	0,068
0,191	0,805	-0,93	-0,83	-0,16	-0,65	1,225	1,345	0,06	1,205
0,328	1,009	0,939	1,031	1,244	0,595	-1,01	0,824	0,231	0,321
-0,23	1,408	1,196	0,855	1,557	1,199	1,243	1,278	0,21	-0,89
-0,74	-0,13	0,783	0,617	-0,34	-0,7	0,33	0,966	1,594	-0,86
0,174	0,408	0,416	-0,07	-0,58	-0,14	0,798	-1,34	1,972	0,197
1,711	1,025	1,099	1,215	1,49	0,924	2,287	0,727	0,885	0,798

Tabel 3.28. Konvolusi pertama - *Output* matriks B *filter* ke-2

Output matriks B *filter* ke-2									
-0,08	-1,05	-0,81	-1,21	-0,32	-0,86	-1,38	-1,06	-0,85	-1,25
0,097	-0,65	0,126	-1,08	-0,19	-0,28	0,101	-1,2	-0,86	-0,39
-0,32	0,059	0,217	-1,27	-0,22	-0,49	-0,85	-0,66	-1,55	-0,78
-1,49	-0,82	-0,13	-0,62	0,39	-0,39	0,087	-0	-1,76	-0,98
-0,64	-0,47	-1,4	-1,57	-0,98	-0,77	-0,42	0,08	-1,26	-0,48
-0,45	0,431	-0,62	-0,95	-0,25	-0,75	-0,7	-0,31	-1,4	-0,66
-0,43	0,011	-0,03	-0,31	0,479	0,426	-0,52	-0,11	-0,64	-1,23
-1,02	-0,3	-0,6	-0,61	-0,19	-0,85	-0,23	-0,46	0,005	-1,63
-0,25	-0,9	-0,58	-0,31	-0,69	-1,49	-0,76	-1,75	0,119	-0,92
0,417	0,346	0,132	0,17	0,273	-0,33	0,713	-0,6	0,514	-0,06

Tabel 3.29. Konvolusi pertama - *Output* matriks B *filter* ke-3

Output matriks B *filter* ke-3									
-0,34	0,251	0,123	-0,19	-1,1	-1,2	-1,41	-1,23	-0,72	0,186
-0,25	-1,04	-0,88	0,199	1,083	1,519	0,949	-0,11	1,524	-0,09
1,342	1,808	0,021	-1,06	-1,48	-1,21	-1,16	-0,22	-0,33	0,521
-0,68	-0,55	1,209	1,812	1,532	0,495	0,549	-0,54	-0,6	-0,33
0,098	-0,98	-0,7	-0,45	-0,32	1,035	0,79	0,418	0,334	-0,06
-0,31	-0,17	-0,13	-0,78	-0,91	-1,74	-0,99	0,163	0,367	-0,04
0,385	0,659	0,118	-0,06	0,594	0,341	-0,33	-0,35	-1,56	-0,23
-0,11	0,711	0,639	0,758	-0,19	0,785	1,668	1,532	0,658	0,25
-0,61	-1,34	-0,42	-1,15	-0,53	0,326	-1,19	-0,63	-0,12	-0,76
-0,15	0,084	-0,49	-0,74	-0,77	-1,31	-1,19	-1,22	-0,46	-0,41

Tabel 3.30. Konvolusi pertama - *Output* matriks B *filter* ke-4

Output matriks B *filter* ke-4									
0,655	-0,35	-0,84	-1,7	0,566	-0,23	-1,18	0,389	0,062	-2,08
0,148	-0,84	1,469	-0,01	-0,78	-0,74	0,288	-1,18	-2,36	-0,54
0,244	-0,49	-0,69	-1,94	0,078	0,658	0,014	-0,54	-0,62	-1,53
-1,2	0,128	0,941	-1,15	-0,41	-1,01	-0,94	1,178	-2,39	-2,04
-0,32	-0,39	-1,17	-0,82	-0,47	-0,47	-0,23	-0,26	-1,48	-1,39
-0,23	0,737	-0,36	-1,61	-0,64	-0,27	0,242	0,606	-1,97	-1,33
0,102	-0,64	-0,12	-0,44	-0,18	0,419	-1,3	-0,84	0,535	-1,68
-0,54	0,177	-0,44	-1,86	0,536	-0,59	-0,09	-0,71	-0,87	-2,49
0,318	-0,56	-1,16	0,967	0,053	-1,78	-0,11	-1,49	0,08	-1,1
0,417	-0,32	0,224	-0,54	-0,01	0,16	0,724	-1	0,441	-0,02

Tabel 3.31. Konvolusi pertama - *Output* matriks B *filter* ke-5

Output matriks B *filter* ke-5									
0,679	0,382	-0,17	-1,32	0,606	0,216	-0,8	0,617	0,898	-1,77
-0,29	-1,01	0,163	0,536	-0,22	-1,62	-0,61	0,766	-2,45	-0,44
-0,37	-1,24	-0,59	-1,01	-0,33	-0,03	0,732	-1,31	-0,12	0,003
-0,62	0,551	0,01	-1,4	-0,76	-0,45	-1,9	0,035	-0,62	-0,78
-0,93	0,513	-0,5	-0,07	0,463	-1	-0,45	-0,95	-1,36	-0,31
-0,58	-0,3	-1,06	-0,84	-0,91	-0,17	0,41	-0,23	-1,39	0,098
-0,01	-1,13	-1,31	-0,51	-0,9	-0,19	-0,7	-1,88	0,632	0,088
-0,26	-0,53	-0,22	-1,34	1,015	-0,11	-1,17	-0,73	-0,55	-0,94
-0,69	0,24	-1,49	1,023	0,812	-1,7	0,059	-0,39	-0,95	-0,9
-0,54	-1,42	0,036	-0,07	-1,65	-0,95	-0,26	-0,95	-0,3	0,284

Tabel 3.32 - 3.37 merupakan *output* matriks RGB yang telah dijumlahkan antara tabel 3.14 - 3.19 berisi *output* matriks R, tabel 3.20 - 3..25 berisi *output* matriks G, tabel 3.26 - 3.31 berisi *output* matriks B dan bias sebanyak 6 berukuran 1x1.

Tabel 3.32. Konvolusi pertama - *Output* matriks RGB *filter* ke-0

Output 0									
-1,061	-1,16	-0,792	0,701	-0,046	0,675	0,839	-3,032	-0,861	2,44
0,853	0,078	-3,04	-3,117	-2,074	-1,57	-1,769	-1,717	-1,632	-1,37
-3,299	-0,253	-1,323	0,659	-0,333	-1,061	-0,604	-2,864	-0,74	-0,692
-0,297	-1,775	-1,267	-2,336	-4,099	-0,805	-1,132	-3,828	-0,17	0,178
-2,062	0,443	-1,847	-1,193	-2,195	-1,583	-0,818	-2,864	-3,196	0,058
-1,886	-2,152	-0,397	-0,497	-2,359	0,834	-2,108	-1,485	-2,25	-0,991
-1,946	-1,999	-0,609	-1,011	-2,764	-3,764	0,808	-1,222	-1,152	-0,821
-0,874	-2,683	-2,607	-0,245	-2,263	0,509	-3,634	-2,35	-2,709	0,873
-1,551	-3,125	1,133	-0,684	-1,404	-0,654	-1,63	0,43	-2,224	0,729
-1,022	-0,142	-1,307	-0,532	0,582	-0,618	-2,977	-0,853	-0,481	-0,291

Tabel 3.33. Konvolusi pertama - *Output* matriks RGB *filter* ke-1

Output 1									
-0,294	-2,459	-1,049	-1,555	-1,719	-1,173	-2,125	-3,306	-1,399	0,404
1,423	-1,102	-1,979	-2,239	-1,603	0,561	-0,879	-2,663	-1,852	0,818
-1,618	0,384	-0,904	-1,066	0,299	-1,691	-1,944	-1,304	-2,16	0,128
-3,464	-2,151	0,028	0,213	-1,109	0,757	-0,31	-0,825	-0,644	0,188
-2,049	0,298	-2,697	-0,364	-2,933	-2,534	-0,176	-0,512	-1,204	3,167
-1,091	0,259	0,593	-0,506	-0,88	-0,101	-1,76	-0,558	-2,868	1,418
-2,557	-1,552	1,337	-0,327	-0,906	-1,647	1,594	0,732	0,045	-1,027
-1,512	-2,364	-0,067	-1,175	-2,457	-2,591	-1,75	0,754	-0,175	0,187
-1,503	-1,743	-0,904	-1,419	-1,548	-0,984	0,178	-1,836	-0,881	1,43
1,69	-0,408	2,272	1,903	0,751	1,447	1,287	0,979	1,888	1,331

Tabel 3.34. Konvolusi pertama - *Output* matriks RGB *filter* ke-2

Output 2									
1,13	0,24	0,486	0,387	-0,059	-0,144	-2,381	-0,196	1,11	-1,696
0,959	0,218	1,198	-0,193	2,246	1,287	-7E-04	-2,054	-0,967	-0,056
1,672	0,503	0,702	-0,535	0,719	-0,825	-0,487	0,549	-0,715	-1,546
-0,856	-0,238	0,827	-0,853	1,849	0,324	-0,214	1,413	-0,459	-0,742
0,978	-0,629	-1,482	-0,771	-1,447	-0,03	-1,324	-0,58	0,271	-0,409
1,044	1,669	-0,662	-1,237	-0,105	-0,491	-0,67	-0,718	-0,663	-0,072
-1,121	1,644	1,262	-0,314	0,002	1,1	0,261	1,188	-0,131	-0,853
0,622	1,286	1,441	-0,446	1,002	-1,293	-0,674	0,897	1,178	-0,589
1,01	0,752	2,34	-0,292	0,255	-1,058	1,029	-1,652	1,48	0,205
1,513	1,279	0,626	0,513	-0,653	-0,492	0,913	1,573	1,547	0,105

Tabel 3.35. Konvolusi pertama - *Output* matriks RGB *filter* ke-3

Output 3									
-1,476	-2,34	-2,038	-1,744	-3,416	-1,246	-4,776	-1,376	-2,786	-0,742
-2,231	-3,702	-4,202	-1,611	-1,264	0,554	-1,84	-1,416	-2,26	-0,659
-0,512	-0,667	-2,84	-3,521	-3,664	-2,832	-3,08	-1,594	-3,346	0,479
-3,871	-1,782	0,619	-0,739	-0,558	-2,145	-0,964	-2,113	-2,764	-0,228
-3,307	-3,423	-0,621	-2,228	-2,333	-1,348	-1,222	-1,952	-1,607	-0,171
-3,814	-1,572	-1,333	-2,585	-3,027	-3,513	-1,117	-2,762	-2,596	-1,171
-3,841	-1,664	-2,457	-1,399	-3,059	-2,286	-1,465	-1,518	-4,362	-0,704
-2,394	-2,346	-1,4	-0,24	-2,265	-2,289	0,67	-1,354	-1,853	-1,103
-4,289	-5,444	-1,721	-4,255	-1,182	-1,387	-1,591	-3,128	-1,537	-2,464
-3,107	-2,908	-1,304	-2,947	-1,562	-2,736	-2,783	-2,89	-1,703	-0,861

Tabel 3.36. Konvolusi pertama - *Output* matriks RGB *filter* ke-4

Output 4									
0,049	-0,106	-0,819	-3,1	0,447	-1,739	-1,647	-1,629	-1,328	-2,256
0,493	-2,46	0,488	-0,637	-2,482	-2,846	0,215	-1,93	-3,676	-1,683
-0,987	-1,613	-3,646	-1,712	0,725	-0,051	-0,945	-2,624	-0,181	-1,981
-1,003	-1,277	0,027	-2,91	-2,013	-0,658	-2,371	-1,015	-2,711	-3,944
-0,406	0,136	-3,559	-0,982	-2,282	-1,962	-1,202	-1,681	-1,413	-1,228
-1,805	0,842	-0,796	-2,51	-2,085	0,623	-1,639	0,166	-3,79	-1,335
-0,461	-1,701	-0,13	-0,562	-1,62	-1,293	-1,181	-2,155	0,554	-2,08
0,439	-0,885	-1,801	-2,982	-1,294	-0,97	-0,941	-1,533	-1,859	-2,215
-1,111	-2,213	-2,166	0,36	-1,705	-3,162	-0,828	-0,682	-2,501	-1,53
-0,136	-0,848	1,032	-0,404	-0,802	0,029	-2,047	-1,226	-0,198	0,019

Tabel 3.37. Konvolusi pertama - *Output* matriks RGB *filter* ke-5

Output 5									
0,959	1,777	-1,167	-1,628	1,016	-0,78	-0,555	0,254	1,243	-1,145
-0,637	-2,8	-1,705	-0,193	-1,587	-2,312	-1,14	-1,465	-3,414	0,636
0,132	-2,192	-1,56	-3,361	-3,558	0,119	-2,385	-3,399	-1,949	-0,527
-2,065	-0,188	-0,268	-1,353	-3,285	-2,722	-4,454	-1,192	-1,513	-1,332
-2,507	-1,646	-1,343	-0,997	-1,247	-1,276	-1,819	-3,67	-3,676	-0,92
-2,08	-1,147	-2,443	-0,189	-2,716	-1,848	-0,159	-0,199	-3,472	0,446
-0,45	-2,87	-2,181	-1,767	-3,402	-1,428	-2,36	-1,677	-0,464	1,181
-0,942	-3,144	-0,858	-2,158	-0,487	-1,404	-2,042	-2,418	0,111	-2,082
-2,74	-1,64	-1,402	-0,573	-1,938	-2,384	-3,335	-0,633	-3,005	0,466
-0,403	-2,652	-1,786	-1,265	-2,166	-2,288	-1,776	-1,878	-2,354	2,224

6) ReLU

Setelah proses konvolusi dilakukan, selanjutnya akan dilakukan proses ReLU, jika *output* hasil proses *convolutional layer* memiliki nilai < 0 maka nilainya berubah menjadi 0, dan jika *output* ini memiliki nilai ≥ 0 maka nilai

tersebut tidak akan berubah. Tabel 3.38 - 3.43 di bawah ini merupakan *output* dari proses konvolusi sebelumnya dan telah diaktivasi dengan ReLU.

Tabel 3.38. ReLU - *output* matriks RGB *filter* ke-0

Output 0 - relu									
0	0	0	0,701	0	0,675	0,839	0	0	2,44
0,853	0,078	0	0	0	0	0	0	0	0
0	0	0	0,659	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0,178
0	0,443	0	0	0	0	0	0	0	0,058
0	0	0	0	0	0,834	0	0	0	0
0	0	0	0	0	0	0,808	0	0	0
0	0	0	0	0	0,509	0	0	0	0,873
0	0	1,133	0	0	0	0	0,43	0	0,729
0	0	0	0	0,582	0	0	0	0	0

Tabel 3.39. ReLu - *output* matriks RGB *filter* ke-1

Output 1 - relu									
0	0	0	0	0	0	0	0	0	0,404
1,423	0	0	0	0	0,561	0	0	0	0,818
0	0,384	0	0	0,299	0	0	0	0	0,128
0	0	0,028	0,213	0	0,757	0	0	0	0,188
0	0,298	0	0	0	0	0	0	0	3,167
0	0,259	0,593	0	0	0	0	0	0	1,418
0	0	1,337	0	0	0	1,594	0,732	0,045	0
0	0	0	0	0	0	0	0,754	0	0,187
0	0	0	0	0	0	0,178	0	0	1,43
1,69	0	2,272	1,903	0,751	1,447	1,287	0,979	1,888	1,331

Tabel 3.40. ReLu - *output* matriks RGB *filter* ke-2

Output 2 - relu									
1,13	0,24	0,486	0,387	0	0	0	0	1,11	0
0,959	0,218	1,198	0	2,246	1,287	0	0	0	0
1,672	0,503	0,702	0	0,719	0	0	0,549	0	0
0	0	0,827	0	1,849	0,324	0	1,413	0	0
0,978	0	0	0	0	0	0	0	0,271	0
1,044	1,669	0	0	0	0	0	0	0	0
0	1,644	1,262	0	0,002	1,1	0,261	1,188	0	0
0,622	1,286	1,441	0	1,002	0	0	0,897	1,178	0
1,01	0,752	2,34	0	0,255	0	1,029	0	1,48	0,205
1,513	1,279	0,626	0,513	0	0	0,913	1,573	1,547	0,105

Tabel 3.41. ReLu - *output* matriks RGB *filter* ke-3

Output 3 - relu									
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0,554	0	0	0	0
0	0	0	0	0	0	0	0	0	0,479
0	0	0,619	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0,67	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabel 3.42. ReLu - *output* matriks RGB *filter* ke-4

Output 4 - relu									
0,049	0	0	0	0,447	0	0	0	0	0
0,493	0	0,488	0	0	0	0,215	0	0	0
0	0	0	0	0,725	0	0	0	0	0
0	0	0,027	0	0	0	0	0	0	0
0	0,136	0	0	0	0	0	0	0	0
0	0,842	0	0	0	0,623	0	0,166	0	0
0	0	0	0	0	0	0	0	0,554	0
0,439	0	0	0	0	0	0	0	0	0
0	0	0	0,36	0	0	0	0	0	0
0	0	1,032	0	0	0,029	0	0	0	0,019

Tabel 3.43. ReLu - *output* matriks RGB *filter* ke-5

Output 5 - relu									
0,959	1,777	0	0	1,016	0	0	0,254	1,243	0
0	0	0	0	0	0	0	0	0	0,636
0,132	0	0	0	0	0,119	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0,446
0	0	0	0	0	0	0	0	0	1,181
0	0	0	0	0	0	0	0	0,111	0
0	0	0	0	0	0	0	0	0	0,466
0	0	0	0	0	0	0	0	0	2,224

7) Perancangan *Output Layer*

Output layer pada sistem ini berupa *convolutional layer* dengan jumlah *filter* sebanyak 3, ukuran kernel 3x3 dan 1 stride yang akan menghasilkan matriks

citra RGB. Tabel 3.44, 3.46, 3.48, 3.50, 3.52, dan 3.54 merupakan matriks *output* yang telah diberi *padding* bernilai 0 disekelilingnya. Tabel 3.45, 3.47, 3.49, 3.51, 3.53 dan 3.55 merupakan *filter* sebanyak 3 *filter* (sesuai dengan *channel* warna RGB) berukuran 3x3 untuk masing-masing matriks R,G dan B. Tabel 3.56 merupakan bias berukuran 1x1 sebanyak 3.

Tabel 3.44. *Output* citra dalam matriks ke-0

Output 0											
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0,701	0	0,675	0,839	0	0	2,44	0
0	0,853	0,078	0	0	0	0	0	0	0	0	0
0	0	0	0	0,659	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0,178	0
0	0	0,443	0	0	0	0	0	0	0	0,058	0
0	0	0	0	0	0	0,834	0	0	0	0	0
0	0	0	0	0	0	0	0,808	0	0	0	0
0	0	0	0	0	0	0,509	0	0	0	0,873	0
0	0	0	1,133	0	0	0	0	0,43	0	0,729	0
0	0	0	0	0	0,582	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.45. 3 *Filter* untuk *output* matriks ke-0

FIL 0		
1	1	1
1	0	1
0	1	0

FIL 1		
-1	0	-1
0	-1	0
1	0	0

FIL 2		
0	1	0
-1	0	0
0	0	0

Tabel 3.46. *Output* citra dalam matriks ke-1

Output 1											
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0,404	0
0	1,423	0	0	0	0	0,561	0	0	0	0,818	0
0	0	0,384	0	0	0,299	0	0	0	0	0,128	0
0	0	0	0,028	0,213	0	0,757	0	0	0	0,188	0
0	0	0,298	0	0	0	0	0	0	0	3,167	0
0	0	0,259	0,593	0	0	0	0	0	0	1,418	0
0	0	0	1,337	0	0	0	1,594	0,732	0,045	0	0
0	0	0	0	0	0	0	0	0,754	0	0,187	0
0	0	0	0	0	0	0	0,178	0	0	1,43	0
0	1,69	0	2,272	1,903	0,751	1,447	1,287	0,979	1,888	1,331	0
0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.47. 3 *Filter* untuk *output* matriks ke-1

FIL 0		
-1	0	1
-1	0	1
-1	1	0

FIL 1		
-1	0	1
0	0	1
1	0	0

FIL 2		
1	1	1
1	0	0
-1	-1	0

Tabel 3.48. *Output* citra dalam matriks ke-2

Output 2											
0	0	0	0	0	0	0	0	0	0	0	0
0	1,13	0,24	0,486	0,387	0	0	0	0	1,11	0	0
0	0,959	0,218	1,198	0	2,246	1,287	0	0	0	0	0
0	1,672	0,503	0,702	0	0,719	0	0	0,549	0	0	0
0	0	0	0,827	0	1,849	0,324	0	1,413	0	0	0
0	0,978	0	0	0	0	0	0	0	0,271	0	0
0	1,044	1,669	0	0	0	0	0	0	0	0	0
0	0	1,644	1,262	0	0,002	1,1	0,261	1,188	0	0	0
0	0,622	1,286	1,441	0	1,002	0	0	0,897	1,178	0	0
0	1,01	0,752	2,34	0	0,255	0	1,029	0	1,48	0,205	0
0	1,513	1,279	0,626	0,513	0	0	0,913	1,573	1,547	0,105	0
0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.49. 3 *Filter* untuk *output* matriks ke-2

FIL 0		
-1	-1	1
-1	1	0
-1	-1	0

FIL 1		
-1	1	-1
0	0	-1
0	0	1

FIL 2		
0	-1	0
1	1	-1
1	1	-1

Tabel 3.50. *Output* citra dalam matriks ke-3

Output 3											
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0,554	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0,479	0
0	0	0	0,619	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0,67	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.51. 3 *Filter* untuk *output* matriks ke-3

FIL 0		
-1	-1	1
0	-1	-1
0	0	0

FIL 1		
0	-1	0
0	1	-1
-1	0	1

FIL 2		
1	1	0
1	-1	-1
0	-1	-1

Tabel 3.52. *Output* citra dalam matriks ke-4

Output 4											
0	0	0	0	0	0	0	0	0	0	0	0
0	0,049	0	0	0	0,447	0	0	0	0	0	0
0	0,493	0	0,488	0	0	0	0,215	0	0	0	0
0	0	0	0	0	0,725	0	0	0	0	0	0
0	0	0	0,027	0	0	0	0	0	0	0	0
0	0	0,136	0	0	0	0	0	0	0	0	0
0	0	0,842	0	0	0	0,623	0	0,166	0	0	0
0	0	0	0	0	0	0	0	0	0,554	0	0
0	0,439	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0,36	0	0	0	0	0	0	0
0	0	0	1,032	0	0	0,029	0	0	0	0,019	0
0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.53. 3 *Filter* untuk *output* matriks ke-4

FIL 0		
0	-1	1
1	1	0
0	-1	0

FIL 1		
0	1	-1
0	0	-1
1	1	1

FIL 2		
0	0	0
-1	0	-1
1	1	0

Tabel 3.54. *Output* citra dalam matriks ke-5

Output 5											
0	0	0	0	0	0	0	0	0	0	0	0
0	0,959	1,777	0	0	1,016	0	0	0,254	1,243	0	0
0	0	0	0	0	0	0	0	0	0	0,636	0
0	0,132	0	0	0	0	0,119	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0,446	0
0	0	0	0	0	0	0	0	0	0	1,181	0
0	0	0	0	0	0	0	0	0	0,111	0	0
0	0	0	0	0	0	0	0	0	0	0,466	0
0	0	0	0	0	0	0	0	0	0	2,224	0
0	0	0	0	0	0	0	0	0	0	0	0

Tabel 3.55. 3 *Filter* untuk *output* matriks ke-5

FIL 0		
-1	-1	0
1	-1	0
0	-1	-1

FIL 1		
1	1	0
0	1	1
-1	1	1

FIL 2		
0	0	1
0	0	1
1	1	0

Tabel 3.56. Bias *convolutional layer* kedua

bias 0
0

bias 1
1

bias 2
-1

Tabel 3.57 - 3.59 merupakan tabel matriks *output* (yang disebut sebagai data prediksi) dari proses konvolusi kedua yang telah digabungkan sehingga menjadi matriks *output* R,G dan B.

Tabel 3.57. Data prediksi - citra dalam matriks *output* R

Matriks output R									
1,045	-4,182	0,82	-1,297	-1,826	-0,672	-1,389	0,585	2,328	0,314
-2,174	-6,229	-1,458	-0,479	0,514	-2,151	-0,119	1,361	0,506	-0,421
1,621	-1,022	-2,178	0,002	0,197	-5,699	-3,58	-0,864	-1,495	-0,749
-1,896	-2,569	-0,781	0,87	1,607	-2,661	-0,65	0,864	-1,259	2,474
0,674	-2,635	-2,315	-0,22	-0,981	-1,962	0,332	-1,579	2,004	0,879
1,202	-0,255	-2,509	-3,192	0,831	-0,48	2,498	-1,875	0,928	-1,886
0,665	-1,888	-5,037	-4,633	0,457	3,219	0,892	-0,234	-2,889	-1,237
1,696	0,296	-4,709	-6,737	2,354	-0,365	1,109	-1,515	-3,945	-2,285
1,411	-4,074	-1,803	-2,794	-0,962	0,768	1,875	-4,39	-2,247	-6,61
1,255	2,059	0,684	-1,333	-1,223	2,099	-0,124	1,966	1,64	-6,958

Tabel 3.58. Data prediksi - citra dalam matriks *output* G

Matriks output G									
4,207	6,746	1,178	3,601	3,857	0,541	0,637	1,602	3,283	-0,803
2,462	1,166	2,318	-0,348	0,921	2,659	0,756	-0,695	3,504	1,769
2,169	-2,931	2,417	-2,963	3,174	-0,61	0,988	1	1,467	2,116
2,821	-2,049	1,866	-2,697	2,219	0,101	-0,843	1,82	0,767	0,342
3,673	0,573	2,549	-1,111	3,692	0,098	0,129	2,307	3,376	1,389
1,09	2,013	0,259	2,34	1,477	0,428	2,022	3,685	8,711	2,955
-0,127	5,016	-0,928	1,406	-1,557	3,626	0,142	1,276	4,464	2,516
-1,178	3,619	0,977	-0,856	0,262	2,472	0,232	-0,393	1,035	1,729
1,303	1,231	1,207	1,607	3,425	1,422	2,727	1,642	4,971	3,802
-0,021	-2,117	3,619	-1,617	2,091	0,268	1,004	-1,346	5,976	2,416

Tabel 3.59 Data prediksi - citra dalam matriks *output* B

Matriks output B									
1,477	-1,116	1,243	-0,119	-0,91	0,971	-0,478	-1,491	0,11	-0,072
1,688	-0,45	0,674	-1,223	0,55	3,702	2,064	0,576	-1	1,236
1,487	0,309	-0,402	-2,977	-1,194	0,535	-2,28	0,962	1,937	-0,849
-1,311	-1,777	-1,273	-0,444	0,437	1,471	-1,332	-0,408	0,812	-3,289
-0,782	3,302	0,411	-0,733	-1,88	0,056	0,38	-2,518	-0,375	-1,334
-4,791	2,836	1,952	-0,485	-2,721	-0,159	-4,043	-1,877	3,141	3,914
-3,654	-0,528	5,485	2,629	-1,096	1,006	-2,394	-0,354	5,397	1,011
-1,405	-1,857	4,894	3,221	0,614	-0,949	0,648	0,055	5,062	0,945
-2,82	-2,388	0,443	-2,053	-4,248	-4,347	-1,912	-1,961	0,222	2,422
-1,776	1,072	-0,814	1,379	1,132	-1,566	-1,093	1,834	5,616	4,494

Tabel 3.60 - 3.62 merupakan tabel matriks data target. Data target merupakan citra asli yang akan digunakan sebagai pembanding untuk menghitung MSE.

Tabel 3.60. Data target - citra dalam matriks R

Data Target R									
0,621	0,581	0,249	0,031	0,184	0,209	0,625	0,081	0,636	0,689
0,305	0,604	0,031	0,858	0,484	0,811	0,608	0,183	0,084	0,041
0,914	0,858	0,773	0,326	0,137	0,461	0,085	0,953	0,352	0,849
0,56	0,908	0,327	0,45	0,197	0,223	0,014	0,921	0,048	0,7
0,958	0,892	0,53	0,21	0,883	0,967	0,781	0,556	0,184	0,388
0,034	0,917	0,482	0,269	0,2	0,675	0,906	0,126	0,879	0,698
0,511	0,498	0,863	0,174	0,048	0,098	0,681	0,167	0,767	0,081
0,403	0,259	0,138	0,172	0,13	0,814	0,904	0,141	0,373	0,255
0,583	0,747	0,303	0,072	0,656	0,05	0,663	0,673	0,165	0,044
0,497	0,381	0,483	0,624	0,359	0,529	0,912	0,956	0,155	0,546

Tabel 3.61. Data target - citra dalam matriks G

Data Target G									
0,956	0,989	0,672	0,111	0,34	0,365	0,262	0,212	0,782	0,003
0,779	0,5	0,786	0,522	0,768	0,691	0,824	0,167	0,631	0,56
0,804	0,034	0,417	0,596	0,09	0,255	0,839	0,255	0,225	0,482
0,008	0,032	0,613	0,731	0,841	0,501	0,874	0,463	0,198	0,694
0,421	0,184	0,922	0,73	0,666	0,377	0,879	0,297	0,875	0,757
0,604	0,318	0,712	0,89	0,599	0,716	0,917	0,434	0,894	0,884
0,906	0,519	0,405	0,643	0,19	0,622	0,342	0,474	0,982	0,264
0,128	0,086	0,62	0,63	0,934	0,402	0,647	0,644	0,332	0,189
0,142	0,935	0,599	0,649	0,146	0,045	0,912	0,837	0,551	0,23
0,745	0,274	0,126	0,486	0,74	0,352	0,5	0,189	0,202	0,392

Tabel 3.62. Data target - citra dalam matriks B

Data Target B									
0,923	0,01	0,049	0,189	0,012	0,936	0,087	0,916	0,923	0,681
0,985	0,043	0,656	0,395	0,932	0,562	0,947	0,304	0,681	0,835
0,362	0,074	0,554	0,511	0,282	0,459	0,983	0,098	0,561	0,407
0,17	0,143	0,079	0,87	0,695	0,382	0,014	0,78	0,374	0,286
0,818	0,671	0,549	0,522	0,759	0,35	0,866	0,883	0,378	0,959
0,305	0,926	0,675	0,812	0,32	0,811	0,849	0,851	0,007	0,205
0,557	0,174	0,009	0,548	0,977	0,731	0,571	0,379	0,967	0,987
0,814	0,435	0,808	0,546	0,24	0,246	0,303	0,086	0,212	0,743
0,658	0,279	0,371	0,093	0,724	0,761	0,872	0,205	0,622	0,524
0,643	0,33	0,13	0,093	0,476	0,159	0,627	0,461	0,764	0,481

Berikut ini merupakan hasil dari metode MSE yang digunakan untuk menghitung nilai eror dari proses *feed forward* yang telah dilakukan, rumus dan perhitungan hasil MSE adalah:

$$MSE = J(\theta_0, \theta_1) = \frac{1}{MM} \sum_{i=0}^{M} \sum_{j=0}^{M} \left((\theta_0 + \theta_1 x_{ij}) - t_{ij} \right)^2$$

$$\boldsymbol{MSE = 5,661}$$

8) Proses *Back-Propagation*

Setelah seluruh rangkaian proses *feed forward* dilakukan, tahap selanjutnya adalah proses *back propagation* atau *backprop*. Pada proses *backprop*, dibutuhkan turunan parsial dari fungsi eror (dalam penelitian ini MSE) yang digunakan untuk mendapat nilai $Gradient\theta_0$ dan $Gradient\theta_1$ yang akan digunakan sebagai parameter pada algoritma ADAM *backprop*. Seperti pada rumus 2.14 dan rumus 2.15..

Dari perhitungan *feed forward* yang telah dilakukan di tahap sebelumnya dan notasi di atas, didapatkan nilai *gradient*-nya sebagai berikut:

$$Gradient\theta_0 = \frac{\partial J}{\partial \theta_0}(\theta_0, \theta_1) = 1{,}516$$

$$Gradient\theta_1 = \frac{\partial J}{\partial \theta_1}(\theta_0, \theta_1) = 0{,}786$$

Berikut ini adalah inisialisasi dari parameter masukkan yang digunakan di dalam algoritma ADAM *backprop*:

- $\beta_1 = 0{,}5$
- $\beta_2 = 0{,}999$
- $\alpha = 0{,}0002$
- $\varepsilon = 0{,}000001$
- $T, m, v = 0$

Berikut ini adalah langkah-langkah dari algoritma ADAM *backprop* untuk proses *backprop* dari *output layer* ke *hidden layer*:

- Hitung *timestep* dengan rumus:

$$T = 0$$

$$T = T + 1$$

- Untuk $Gradient\theta_0$ hitung $m, v, \widehat{m}$, dan $\hat{v}$ dengan rumus:

$$m_b = \beta_1 . m_b + (1 - \beta_1) . Gradient\theta_0$$

$$m_b = 0{,}758$$

$$v_b = \beta_2 . v_b + (1 - \beta_2) . Gradient{\theta_0}^2$$

$$v_b = 0{,}002$$

$$\widehat{m_b} = \frac{m_b}{1 - (\beta_1^T)}$$

$$\widehat{m_b} = 1{,}516$$

$$\widehat{v_b} = \frac{v_b}{1 - (\beta_2^T)}$$

$$\widehat{v_b} = 1{,}516$$

- Untuk $Gradient\theta_1$ hitung $m, v, \widehat{m}$, dan $\hat{v}$ dengan rumus:

$$m_f = \beta_1 . m_f + (1 - \beta_1) . Gradient\theta_1$$

$$m_f = 0{,}393$$

$$v_f = \beta_2 . v_f + (1 - \beta_2) . Gradient{\theta_1}^2$$

$$v_f = 0{,}0008$$

$$\widehat{m_f} = \frac{m_f}{1 - (\beta_1^T)}$$

$$\widehat{m_f} = 0{,}786$$

$$\widehat{v_f} = \frac{v_f}{1 - (\beta_2^T)}$$

$$\widehat{v_f} = 0{,}786$$

- *Update* bias dengan rumus (2.17)

  Tabel 3.63 dibawah ini merupakan tabel bias yang sudah di-*update*.

  Tabel 3.63. Nilai bias baru

Nbias 0	Nbias 1	Nbias 2
0,00	0,81	-0,81

- *Update filter* dengan rumus (2.19)

Tabel 3.64 hingga tabel 3.69 di bawah ini merupakan tabel *filter* yang sudah di-*update*.

Tabel 3.64. Nilai *filter* baru untuk *output* matriks ke-0

Nfilter 0		
1,1276	1,1276	1,1276
1,1276	-0,0002	1,1276
-0,0002	1,1276	-0,0002

Nfilter 1		
-1,1279	-0,0002	-1,1279
-0,0002	-1,1279	-0,0002
1,1276	-0,0002	-0,0002

Nfilter 2		
-0,0002	1,1276	-0,0002
-1,1279	-0,0002	-0,0002
-0,0002	-0,0002	-0,0002

Tabel 3.65. Nilai *filter* baru untuk *output* matriks ke-1

Nfilter 0		
-1,1279	-0,0002	1,1276
-1,1279	-0,0002	1,1276
-1,1279	1,1276	-0,0002

Nfilter 1		
-1,1279	-0,0002	1,1276
-0,0002	-0,0002	1,1276
1,1276	-0,0002	-0,0002

Nfilter 2		
1,1276	1,1276	1,1276
1,1276	-0,0002	-0,0002
-1,1279	-1,1279	-0,0002

Tabel 3.66. Nilai *filter* baru untuk *output* matriks ke-2

Nfilter 0		
-1,1279	-1,1279	1,1276
-1,1279	1,1276	-0,0002
-1,1279	-1,1279	-0,0002

Nfilter 1		
-1,1279	1,1276	-1,1279
-0,0002	-0,0002	-1,1279
-0,0002	-0,0002	1,1276

Nfilter 2		
-0,0002	-1,1279	-0,0002
1,1276	1,1276	-1,1279
1,1276	1,1276	-1,1279

Tabel 3.67. Nilai *filter* baru untuk *output* matriks ke-3

Nfilter 0		
-1,1279	-1,1279	1,1276
-0,0002	-1,1279	-1,1279
-0,0002	-0,0002	-0,0002

Nfilter 1		
-0,0002	-1,1279	-0,0002
-0,0002	1,1276	-1,1279
-1,1279	-0,0002	1,1276

Nfilter 2		
1,1276	1,1276	-0,0002
1,1276	-1,1279	-1,1279
-0,0002	-1,1279	-1,1279

Tabel 3.68. Nilai *filter* baru untuk *output* matriks ke-4

Nfilter 0		
-0,0002	-1,1279	1,1276
1,1276	1,1276	-0,0002
-0,0002	-1,1279	-0,0002

Nfilter 1		
-0,0002	1,1276	-1,1279
-0,0002	-0,0002	-1,1279
1,1276	1,1276	1,1276

Nfilter 2		
-0,0002	-0,0002	-0,0002
-1,1279	-0,0002	-1,1279
1,1276	1,1276	-0,0002

Tabel 3.69. Nilai *filter* baru untuk *output* matriks ke-5

Nfilter 0		
-1,1279	-1,1279	-0,0002
1,1276	-1,1279	-0,0002
-0,0002	-1,1279	-1,1279

Nfilter 1		
1,1276	1,1276	-0,0002
-0,0002	1,1276	1,1276
-1,1279	1,1276	1,1276

Nfilter 2		
-0,0002	-0,0002	1,1276
-0,0002	-0,0002	1,1276
1,1276	1,1276	-0,0002

Setelah proses perhitungan *backprop* dari *output layer* ke *hidden layer* dilakukan, didapatlah nilai bias dan *filter* yang baru. Selanjutnya adalah proses perhitungan *backprop* dari *hidden layer* ke *input layer*.

- Lakukan proses konvolusi dengan nilai *filter* baru dan nilai bias baru. Tabel 3.70 - 3.72 merupakan tabel hasil konvolusi antara matriks *output* (tabel 3.57 - 3.59) dengan masing-masing *filter* (tabel 3.64 - 3.69) dan bias (tabel 3.63). Hasil konvolusi setiap *filter* dijumlahkan sehingga didapatkan 3 matriks R, G dan B.

Tabel 3.70. Citra dalam matriks R

Matriks R									
1,177	-4,718	0,924	-1,463	-2,061	-0,759	-1,567	0,659	2,624	0,353
-2,453	-7,027	-1,646	-0,541	0,578	-2,428	-0,136	1,534	0,569	-0,476
1,827	-1,154	-2,457	-1E-04	0,221	-6,429	-4,038	-0,974	-1,686	-0,846
-2,139	-2,898	-0,881	0,98	1,812	-3,002	-0,733	0,973	-1,421	2,79
0,759	-2,973	-2,612	-0,248	-1,108	-2,213	0,374	-1,78	2,258	0,99
1,355	-0,29	-2,831	-3,601	0,937	-0,542	2,816	-2,115	1,045	-2,128
0,748	-2,131	-5,682	-5,226	0,515	3,628	1,004	-0,265	-3,26	-1,396
1,911	0,331	-5,312	-7,599	2,654	-0,413	1,249	-1,711	-4,451	-2,579
1,59	-4,598	-2,036	-3,153	-1,086	0,865	2,113	-4,953	-2,537	-7,457
1,414	2,32	0,769	-1,506	-1,38	2,366	-0,142	2,215	1,847	-7,848

Tabel 3.71. Citra dalam matriks G

Matriks G									
3,616	6,478	0,2	2,932	3,221	-0,519	-0,41	0,678	2,573	-2,035
1,647	0,185	1,485	-1,522	-0,091	1,869	-0,276	-1,912	2,823	0,866
1,317	-4,435	1,597	-4,471	2,45	-1,817	-0,015	-4E-04	0,526	1,258
2,053	-3,44	0,976	-4,17	1,373	-1,014	-2,079	0,925	-0,264	-0,742
3,013	-0,483	1,746	-2,382	3,035	-1,019	-0,983	1,474	2,679	0,437
0,1	1,141	-0,837	1,51	0,538	-0,646	1,151	3,027	8,694	2,203
-1,272	4,527	-2,176	0,457	-2,884	2,961	-0,969	0,31	3,905	1,708
-2,458	2,951	-0,028	-2,094	-0,833	1,659	-0,867	-1,573	0,037	0,821
0,34	0,257	0,231	0,682	2,734	0,475	1,946	0,721	4,475	3,157
-1,152	-3,517	2,951	-2,953	1,229	-0,826	0,003	-2,647	5,61	1,595

Tabel 3.72. Citra dalam matriks B

Matriks B									
2,792	-0,132	2,528	0,993	0,1	2,222	0,588	-0,554	1,251	1,046
3,029	0,618	1,887	-0,253	1,746	5,301	3,455	1,777	-0,002	2,52
2,804	1,474	0,673	-2,232	-0,221	1,73	-1,444	2,212	3,312	0,17
-0,351	-0,878	-0,309	0,626	1,619	2,786	-0,375	0,668	2,042	-2,582
0,244	4,85	1,59	0,301	-0,993	1,19	1,555	-1,712	0,704	-0,377
-4,276	4,325	3,328	0,581	-1,941	0,948	-3,433	-0,99	4,668	5,541
-2,994	0,53	7,312	4,092	-0,109	2,261	-1,574	0,727	7,212	2,267
-0,458	-0,969	6,644	4,759	1,819	0,057	1,856	1,187	6,834	2,192
-2,053	-1,568	1,624	-1,19	-3,664	-3,776	-1,03	-1,086	1,375	3,857
-0,876	2,335	0,207	2,681	2,403	-0,64	-0,106	3,194	7,458	6,194

- Pada tahap *feed forward* sebelumnya, terdapat proses ReLU di *layer* ini. Maka, pada tahap *backprop* yang berada di *layer* yang sama harus dilakukan proses ReLU, tepatnya turunan dari fungsi ReLU.

Jika *output* proses konvolusi yang telah dilakukan sebelumnya memiliki nilai ≤ 0 maka hasilnya adalah 0. Dan jika *output* ini memiliki nilai > 0 maka hasilnya adalah 1.

- Setelah itu, lakukan perhitungan MSE seperti sebelumnya.
- Kembali ke langkah awal algoritma *backprop* untuk meng-*update* bias dan *filter* yang berada diantara *hidden layer* dan *input layer*.

3.4. Implementasi Jaringan Syaraf Tiruan

Berikut ini adalah beberapa parameter yang diterapkan pada sistem JST yang digunakan dalam penelitian ini:

1. Inisialisasi *filter*, bias dan *hidden layer* yang digunakan bernilai -1 hingga 1 yang diberikan secara random.
2. *Learning rate* sebesar 0,0002.
3. Fungsi aktivasi yang digunakan yaitu fungsi aktivasi ReLU.
4. Metode *upsampling* yang digunakan yaitu metode *Bilinear upsampling*.
5. Toleransi nilai *error* sebesar 0,000001.
6. Jumlah maksimum *epoch* sebesar 100.

Berikut ini adalah daftar data target yang akan digunakan sebagai tujuan dari data *input*. Data target akan digunakan untuk membandingkan seberapa baik tampilan yang akan dihasilkan oleh sistem.

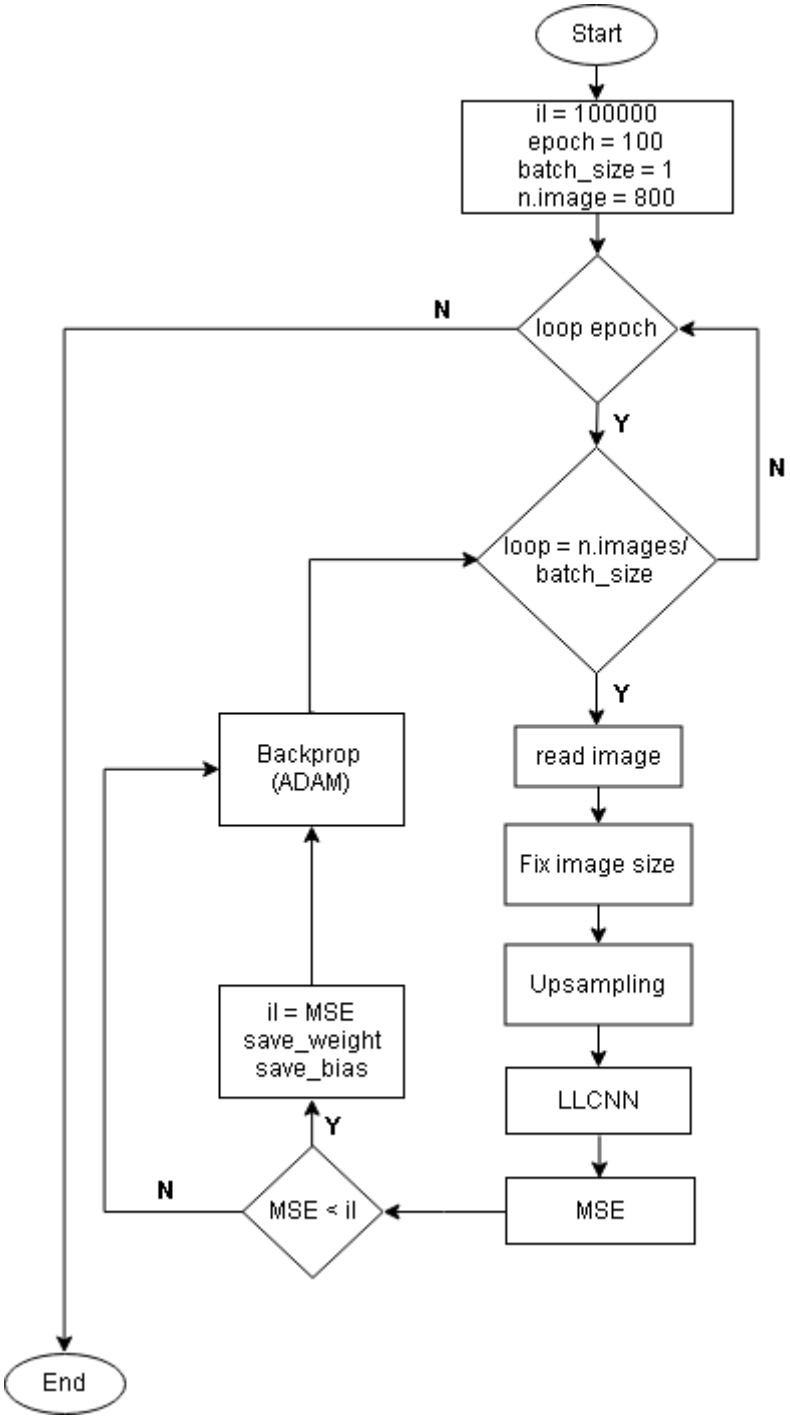
Tabel 3.73. Tabel data *input* dan data target

Data *Input*	Ukuran	Data Target	Ukuran

	170x113 piksel		680x452 piksel
	114x170 piksel		456x680 piksel
	170x170 piksel		680x680 piksel

Untuk data *input* yang memiliki ukuran < 170x170 akan diproses melalui tahapan *pre-processing*. *Pre-processing* akan memproses citra tersebut dengan menambahkan *padding* bernilai 0 pada sisi yang dianggap memiliki ukuran < 170.

Saat seluruh parameter telah ditentukan, tahap selanjutnya yaitu tahap pelatihan. *Data training* yang sudah disiapkan sebelumnya menjadi *input* dan dilatih oleh sistem JST. Proses pelatihan data akan berhenti saat sistem menyentuh toleransi nilai *error* atau mencapai maksimum *epoch*.



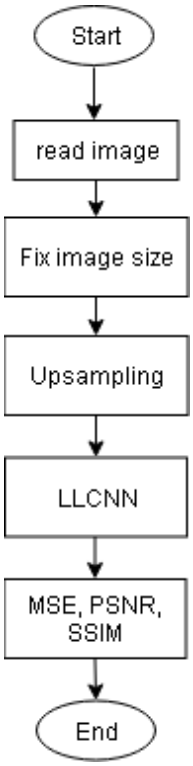
Gambar 3.2. *Flowchart* pelatihan sistem

BAB IV

PENGUJIAN DAN ANALISIS HASIL

4.1. Uji Coba

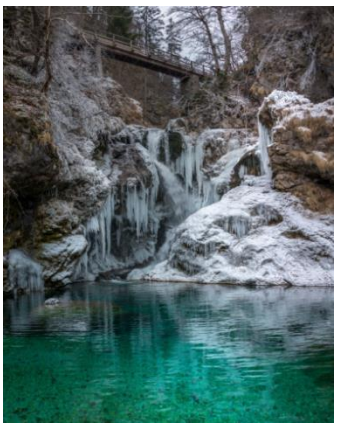
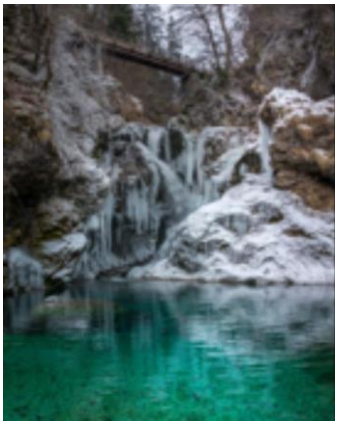
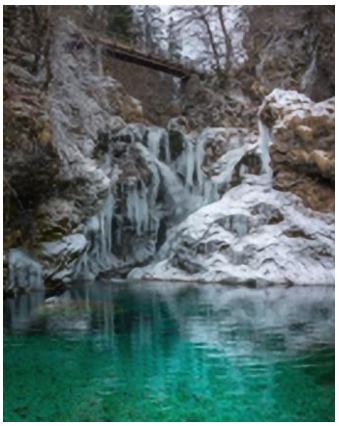
Pada proses uji coba sistem digunakan 200 *data testing*. *Data testing* merupakan data yang belum dikenal oleh sistem, karena data ini tidak digunakan saat proses pelatihan. *Output* hasil proses *testing* akan dibandingkan dengan metode pengujian kuantitatif. Gambar 4.1 merupakan tampilan *flowchart* untuk proses *testing*.

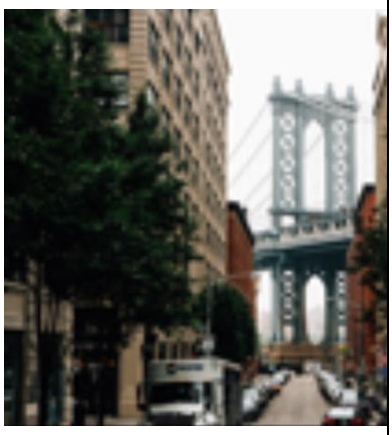


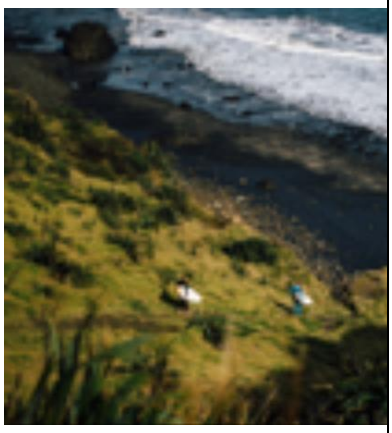
Gambar 4.1. *Flowchart* pengujian sistem

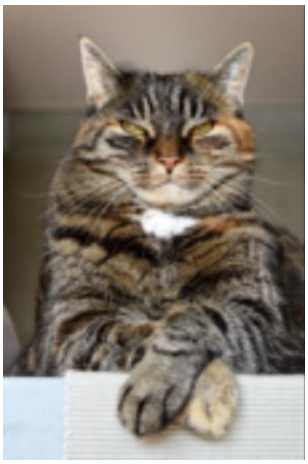
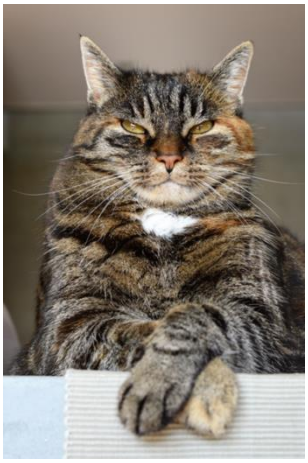
Tabel 4.1. Berikut ini merupakan hasil dari proses pengujian kuantitatif beserta tampilan visual dari hasil proses interpolasi *Bilinear upsampling* dan hasil proses LLCNN:

Tabel 4.1. Tampilan perbandingan citra LLCNN dengan *Bilinear*

No	Data *Input* dan Data Target	Pengujian Visual	Pengujian Kuantitatif
1	 **Input**  **Target**	 ***Bilinear***  **LLCNN**	MSE = 0.00456 SSIM = 0.5782 PSNR = 23.410 MSE = 0.00377 SSIM = 0.6519 PSNR = 24.239

2	 **Input**	 ***Bilinear***	**MSE = 0.00551** **SSIM = 0.6874** **PSNR = 22.586**
	 **Target**	 **LLCNN**	**MSE = 0.00425** **SSIM = 0.7540** **PSNR = 23.709**
3	 **Input**	 ***Bilinear***	**MSE = 0.00192** **SSIM = 0.826** **PSNR = 27.167**

			**MSE = 0.00139** **SSIM = 0.8606** **PSNR = 28.556**
	Target	**LLCNN**	
4			**MSE = 0.00221** **SSIM = 0.7354** **PSNR = 26.554**
	Input	***Bilinear***	
			**MSE = 0.00164** **SSIM = 0.7895** **PSNR = 26.371**
	Target	**LLCNN**	

5	 **Input**	 ***Bilinear***	**MSE = 0.00290** **SSIM = 0.7730** **PSNR = 25.374**
	 **Target**	 **LLCNN**	**MSE = 0.00231** **SSIM = 0.8212** **PSNR = 26.371**

4.2. Analisis Hasil

Dari seluruh rangkaian penelitian yang telah dilakukan, dapat dianalisis bahwa model dari algoritma LLCNN dapat menjadi metode yang dapat menutupi kelemahan dari proses interpolasi *bilinear upsampling*. Metode LLCNN yang diterapkan pada kasus perbaikan citra resolusi rendah ternyata dapat menghasilkan citra resolusi tinggi yang memiliki kualitas piksel yang baik yang dapat dibuktikan pada tabel 4.1 yang berisi tabel pengujian secara kuantitatif dan tampilan visual citra perbandingan antara LLCNN dengan *Bilinear Upsampling*.

BAB V

PENUTUP

5.1. Kesimpulan

Kesimpulan yang didapat dari seluruh rangkaian penelitian dengan judul "Peningkatan Resolusi Citra Digital Menggunakan *Deep Neural Network*" adalah:

- Penelitian yang dilakukan telah berhasil memperbaiki kualitas citra resolusi rendah menjadi citra resolusi tinggi.
- Metode LLCNN yang diterapkan dalam penelitian dapat memperbaiki citra hasil metode *Bilinear upsampling*.

5.2. Saran

Dari permasalahan yang muncul dalam penelitian ini, saran yang dapat dipertimbangkan untuk penelitian selanjutnya, adalah seberapa optimal citra resolusi rendah dapat ditingkatkan ke resolusi tinggi.

DAFTAR PUSTAKA

[1] Tao, Li; et al., "LLCNN: A Convolutional Neural Network for Low-light Image Enhancement," *VCIP 2017,* Vol. %1 dari %2978-1-5386-0462-5/17/$31.00 ©2017 IEEE, 2017.

[2] Lore, Kin Gwn; et al., "LL-Net: A Deep Autoencoder approach to Natural Low-light Image Enhancement," *ArXiv Version,* 2016.

[3] Gooodfellow, Ian; et al., Deep Learning, MIT Press, 2016.

[4] R. Munir, Diktat kuliah Pengolahan Citra, Bandung: Departemen Teknik Informatika ITB, 2002.

[5] Fuad, Nurul; et al., "Analisa Hasil Perbandingan Metode Low-Pass Filter dengan Median Filter Untuk Optimalisasi Kualitas Citra Digital," *Jurnal Teknika,* vol. 4, no. 2085 - 059, pp. 395-396, 2012.

[6] Bertalya, Representasi Citra, Jakarta, 2006.

[7] Endra, "Rekonstruksi Citra dari Pencuplikan Kompresif," *Jurnal Teknik Komputer,* vol. 19, pp. 159-173, 2011.

[8] W., Tjokorda Agung Budi; et al., "Proses Up-Scaling Citra Digital Pada Domain frekuensi dengan menggunakan Metode Discrete Wavelet Transform," *Konferensi Nasional Sistem dan Informatika,* Vol. %1 dari %2KNS&I09-018, p. 96, 2009.

[9] R. C. Gonzalez and R. E. Woods, Digital Image Processing, New Jersey: Prentice Hall, 2001.

[10] S. Marsland, Machine Learning: an algorithmic perspective, Ashhurst, New Zealand: CRC Press, 2011.

[11] A. Ahmad, "Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning," *Jurnal Teknologi Indonesia,* 2017.

[12] J. W. G. Putra, Pengenalan Konsep Pembelajaran Mesin dan deep Learning, 2018.

[13] D. P. Kingma dan J. L. Ba, "ADAM: A Method For Stochastic Optimization," *ICLR,* 2015.

[14] Suartika, I Wayan; et al., “Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101,” *Jurnal Teknik ITS,* vol. 5, pp. A65-A69, 2016.

[15] Sohl-Dickstein, Jascha ; et al., “Fast large-scale optimization by unifying stochastic gradient and quasi-Newton methods,” *In Proceedings of the 31st International Conference on Machine (ICML-14),* pp. 604-612, 2014.

LAMPIRAN

```
import scipy
import scipy.misc
from glob import glob
import numpy as np
#import random
from imageio import imwrite

class DataLoader():
    def __init__(self, hr_res=(2040, 2040), lr_res=(510,510), rgb=True):
        if rgb:
            self.hr_res = (hr_res[0],hr_res[1], 3)
            self.lr_res = (lr_res[0],lr_res[1], 3)
        else:
            self.hr_res = hr_res
            self.lr_res = lr_res

    def load_data(self, data_type="train",  batch_size=1, random=False):
        path_LR = glob('./dataset/%s_LR/*' % (data_type))
        path_HR = glob('./dataset/%s_HR/*' % (data_type))
        #sample_LR = random.sample(path_LR,batch)
        sample_LR = np.random.choice(path_LR, size=batch_size)
        ind = []
        for i in sample_LR:
            ind.append(path_LR.index(i))

        imglr = []
        imghr = []
        Shape = []
        for i in range(batch_size):
            imgl = self.imread(sample_LR[i])
            imgh = self.imread(path_HR[ind[i]])
            imgl = np.array(imgl)
            imgh = np.array(imgh)
            imxl = self.norm_img(imgl, hr=False)
            imxh = self.norm_img(imgh)
            imglr.append(imxl)
            imghr.append(imxh)
            Shape.append(imgl.shape)

        imglr = np.array(imglr)/255
        imghr = np.array(imghr)/255

        return imglr, imghr, Shape

    def load_batch(self, batch_size=1, is_testing=False):
```

```
    data_type = "train" if not is_testing else "val"
    path_LR = glob('./dataset/%s_LR/*' % (data_type))
    path_HR = glob('./dataset/%s_HR/*' % (data_type))

    self.n_batches = int(min(len(path_LR), len(path_HR)) / batch_size)
    total_samples = self.n_batches * batch_size

    # Sample n_batches * batch_size from each path list so that model sees all
    # samples from both domains
    path_A = np.random.choice(path_A, total_samples, replace=False)
    path_B = np.random.choice(path_B, total_samples, replace=False)

    for i in range(self.n_batches-1):
        batch_A = path_A[i*batch_size:(i+1)*batch_size]
        batch_B = path_B[i*batch_size:(i+1)*batch_size]
        imgs_A, imgs_B = [], []
        for img_A, img_B in zip(batch_A, batch_B):
            img_A = self.imread(img_A)
            img_B = self.imread(img_B)

            img_A = scipy.misc.imresize(img_A, self.img_res)
            img_B = scipy.misc.imresize(img_B, self.img_res)

            if not is_testing and np.random.random() > 0.5:
                    img_A = np.fliplr(img_A)
                    img_B = np.fliplr(img_B)

            imgs_A.append(img_A)
            imgs_B.append(img_B)

        imgs_A = np.array(imgs_A)/127.5 - 1.
        imgs_B = np.array(imgs_B)/127.5 - 1.

        yield imgs_A, imgs_B

def norm_img(self, img, hr=True):
    if hr:
        imx = np.zeros(self.hr_res)
    else:
        imx = np.zeros(self.lr_res)
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            imx[i][j] = img[i][j]
    return imx

def save_img(self, path, img, shape=(510,510)):
    row = shape[0]
    col = shape[1]
```

```
    imwrite("HR" + path , img[0:row*4,0:col*4])
    imwrite("LR" + path , img[0:row,0:col])

def imread(self, path):
    return scipy.misc.imread(path, mode='RGB').astype(np.float)

def imsave(self, path, x):
    return scipy.misc.imsave(path, x)
```