TUGAS AKHIR – KS184822

SEGMENTASI TUMOR OTAK PADA CITRA MRI MENGGUNAKAN *FULLY CONVOLUTIONAL NETWORK*

TAUFIK AZMI
NRP 062115 4000 0107

Dosen Pembimbing
Prof. Drs. Nur Iriawan, Mikom., Ph.D

PROGRAM STUDI SARJANA
DEPARTEMEN STATISTIKA
FAKULTAS MATEMATIKA, KOMPUTASI, DAN SAINS DATA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2019

TUGAS AKHIR – KS184822

SEGMENTASI TUMOR OTAK PADA CITRA MRI MENGGUNAKAN *FULLY CONVOLUTIONAL NETWORK*

TAUFIK AZMI
NRP 062115 4000 0107

Dosen Pembimbing
Prof. Drs. Nur Iriawan, Mikom., Ph.D

PROGRAM STUDI SARJANA
DEPARTEMEN STATISTIKA
FAKULTAS MATEMATIKA, KOMPUTASI, DAN SAINS DATA
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2019

FINAL PROJECT – KS184822

MRI-BASED BRAIN TUMOR SEGMENTATION USING FULLY CONVOLUTIONAL NETWORK

TAUFIK AZMI
SN 062115 4000 0107

Supervisor
Prof. Drs. Nur Iriawan, Mikom., Ph.D

UNDERGRADUATE PROGRAMME
DEPARTMENT OF STATISTICS
FACULTY OF MATHEMATICS, COMPUTING, AND DATA SCIENCE
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2019

LEMBAR PENGESAHAN

SEGMENTASI TUMOR OTAK PADA CITRA MRI MENGGUNAKAN *FULLY CONVOLUTIONAL NETWORK*

TUGAS AKHIR
Diajukan untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Statistika
pada
Program Studi Sarjana Departemen Statistika
Fakultas Matematika, Komputasi, dan Sains Data
Institut Teknologi Sepuluh Nopember

Oleh :
Taufik Azmi
NRP. 062115 4000 0107

Disetujui oleh Pembimbing:
Prof. Drs. Nur Iriawan, MIKom., Ph.D
NIP. 19621015 198803 1 002

Mengetahui,
Kepala Departemen Statistika

Dr. Suhartono
NIP. 19710929 199512 1 001

SURABAYA, JULI 2019

SEGMENTASI TUMOR OTAK PADA CITRA MRI MENGGUNAKAN *FULLY CONVOLUTIONAL NETWORK*

Nama Mahasiswa : Taufik Azmi
NRP : 062115 4000 0107
Departemen : Statistika-FMKSD-ITS
Dosen Pembimbing : Prof. Drs. Nur Iriawan, MIKom., Ph.D

Abstrak

Tumor otak adalah pertumbuhan jaringan abnormal pada sel-sel otak yang terus tumbuh dan berlipat ganda tanpa terkendali. Deteksi tumor otak dapat dilakukan dengan pemeriksaan laboratorium dan pemeriksaan radiologis. Magnetic resonance imaging (MRI) adalah salah satu pemeriksaan radiologis yang dapat memindai area otak untuk mendeteksi lokasi tumor otak. Masalah medisnya adalah untuk menentukan tumor otak secara lebih tepat, yang dilakukan dengan memisahkan segmen pada otak (sebagai Region of Interest atau ROI) pada segmen lain dalam MRI. Dalam penelitian ini, klasifikasi ROI dan Non-ROI dilakukan dengan menggunakan Fully Convolutional Network (FCN). Metode ini dapat memproses data dengan pola grid dalam konstruksi matematika. Klasifikasi dengan FCN akan divalidasi dengan menghitung Correct Classification Ratio (CCR) yang dilakukan dengan membandingkan hasilnya dengan ground truth. Hasil klasifikasi dan segmentasi menggunakan FCN dapat mengenali tumor otak pada gambar MRI dengan nilai rata-rata CCR hingga 94,65%.

Kata kunci: Fully Convolutional Network, Magnetic Resonance Imaging, Segmentasi, Tumor Otak

(Halaman ini sengaja dikosongkan)

MRI-BASED BRAIN TUMOR SEGMENTATION USING FULLY CONVOLUTIONAL NETWORK

Name	**: Taufik Azmi**
Student Number	**: 062115 4000 0107**
Department	**: Statistics**
Supervisor	**: Prof. Drs. Nur Iriawan, MIKom., Ph.D**

Abstract

Brain tumors are an abnormal tissue growth in brain cells that continue to grow and multiply uncontrollably. The detection of brain tumor can be done under the laboratory examination and radiological examination. Magnetic Resonance Imaging (MRI) is one of the radiological examinations that could scan the brain area to detect the location of the brain tumor. The medical problem is how to specify the brain tumor more precisely, that is accomplished by separating the brain segment (as the Region of Interest or ROI) to the other segment in the MRI. In this study, the classification of ROI and Non-ROI is done by employing the Fully Convolutional Network (FCN). This method can process data with a grid pattern in a mathematical construction. Classification with FCN would be validated by calculating the Correct Classification Ratio (CCR) which is done by comparing the segmentation results with the ground truth. The results show that the classification and segmentation using FCN could recognize the brain tumor in the MRI image with average CCR value of 94,65%.

Keywords: Brain Tumors, Fully Convolutional Network, Magnetic Resonance Imaging, Segmentation

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur kepada Allah SWT atas limpahan berkat rahmat, karunia, taufik, dan hidayah-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul "**Segmentasi Tumor Otak pada Citra MRI menggunakan *Fully Convolutional Network***".

Tugas Akhir ini dapat terselesaikan dengan baik karena tidak lepas dari bantuan, dukungan, dan peran serta dari berbagai pihak. Oleh karena itu penulis ingin mengucapkan terima kasih kepada.

1. Ayah Saifudin, Ibu Titiek Tjaturweni, kakak serta seluruh keluarga atas segala doa dan dukungan yang diberikan kepada penulis.
2. Bapak Dr. Suhartono selaku Kepala Departemen Statistika yang telah memberikan banyak fasilitas untuk kelancaran penyelesaian Tugas Akhir ini.
3. Bapak Prof. Drs. Nur Iriawan, MIKom., Ph.D. selaku dosen pembimbing yang telah memberikan nasihat, kritik, saran dan waktu yang diberikan kepada penulis dalam menyelesaikan Tugas Akhir ini.
4. Ibu Dra. Wiwiek Setya Winahju, M.Si. dan Ibu Dr. Irhamah, S.Si., M.Si. selaku dosen penguji atas saran dan kritiknya yang sangat membangun.
5. Ibu Dr. Dra. Kartika Fitriasari, M.Si. selaku dosen wali atas nasihat dan saran yang telah diberikan.
6. Ibu Anindya Apriliyanti Pravitasari, S.Si., M.Si., atas bantuan, nasihat, kritik, saran, serta waktu yang diberikan kepada penulis selama penyelesaian Tugas Akhir ini.
7. dr. Widiana Ferriastuti, Sp. Rad (K) atas bantuan, saran, dan waktu yang diberikan kepada penulis dalam pemberian rekomendasi dan validasi dari sudut pandang medis untuk Tugas Akhir ini.
8. Seluruh dosen dan civitas akademika Departemen Statistika ITS yang telah memberikan ilmu yang berharga serta telah membantu kelancaran pelaksanaan perkuliahan.

9. Teman-teman terdekat Ulfa Siti N., Arlandio Nur F., Nur Indah A., M. Ihsan Ananto, Devita Prima V, M. Trianto Utomo, Nur Fidyah P., Cahya Buana P., M. Haidar Alvin P., dan Nursetyo Purwantoro yang telah memberi dukungan dan berdiskusi dalam menyelesaikan Tugas Akhir ini.
10. Teman-teman seperjuangan Tugas Akhir Citra MRI Tumor Otak, Yusuf puji H dan Mbak Nur Indah yang selalu meluangkan waktu untuk berdiskusi bersama selama pengerjaan tugas akhir ini.
11. Semua mahasiswa Statistika S1 ITS angkatan 2015 yang telah menemani, belajar bersama, dan tertawa bersama.
12. Semua pihak yang mendukung dalam penyelesaian Tugas Akhir ini yang tidak dapat penulis sebutkan satu-persatu.

Penulis berharap semoga laporan Tugas Akhir ini dapat bermanfaat dan menambah wawasan bagi pembaca. Kritik dan saran sangat diperlukan untuk perbaikan di masa yang akan datang.

Surabaya, Juli 2019

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

Halaman

(Halaman ini sengaja dikosongkan)

DAFTAR LAMPIRAN

Halaman

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

1.1 Latar Belakang

Otak termasuk salah satu organ penting pada tubuh manusia yang berfungsi sebagai sistem saraf pusat. Secara umum sistem saraf mempunyai tiga fungsi utama yaitu menerima informasi dalam bentuk rangsangan ataupun stimulus, memproses informasi yang diterima dan memberi tanggapan (respon) terhadap rangsang. Jika otak mengalami ganguan maka semua koordinasi pada tubuh juga tergangu. Salah satu jenis penyakit yang dapat menggangu fungsi otak adalah tumor otak (Yueniwati. 2017).

Tumor adalah adanya pertumbuhan jaringan abnormal pada sel yang terus tumbuh dan bermultiplikasi secara tidak terkontrol. Pada saat seseorang terkena tumor otak, pertumbuhan sel yang tidak semestinya menimbulkan penekanan dan kerusakan pada sel-sel lain pada otak dan mengganggu fungsi kerja otak (Yueniwati. 2017). Tercatat bahwa di Amerika Serikat pada 2011 hingga 2015 tingkat penderita tumor otak dan tumor CNS lainnya (baik ganas maupun tidak ganas) merupakan kanker yang paling umum terjadi pada orang berusia 0-14 tahun. Tercatat penderita dengan usia 0-14 tahun memiliki tingkat kejadian rata-rata setiap tahunnya 5,65 per 100.000 penduduk. Pada rentang usia 15-39 tahun tumor otak dan tumor CNS lainnya (baik ganas maupun tidak ganas) memiliki rata-rata kejadian tahunan sebesar 11,20 per 100.000 penduduk. Tumor otak dan tumor CNS lainnya (baik ganas maupun tidak ganas) adalah kanker kedelapan yang paling umum di antara orang berusia 40+ tahun dengan rata-rata kejadian tahunan 44,47 per 100.000 penduduk (Ostrom dkk, 2018).

Menentukan seorang pasien mengidap penyakit tumor otak, seorang dokter tidak dapat memutuskannya hanya berdasarkan pengecekan tanya jawab semata. Diperlukan pengecekan lebih lanjut untuk memastikan seseorang mengidap penyakit tumor otak. Pemeriksaan yang dilakukan untuk menunjang dalam vonis tumor otak dapat dilakukan dengan pemeriksaan laboratorium, radiologi,

dan cairan serebrospinal (Yueniwati. 2017). Salah satu alat yang digunakan dalam pemeriksaan radiologi adalah *Magnetic Resonance Imaging* (MRI). Pemeriksaan MRI bertujuan untuk mengetahui karakteristik morfologik (lokasi, ukuran, bentuk, perluasan dan lain lain) dari keadaan patologis. Tujuan tersebut dapat diperoleh dengan menilai dari salah satu atau kombinasi gambar penampang tubuh *axial*, *sagittal* dan *coronal* serta tergantung pada letak organ dan kemungkinan patologinya. Pemanfatan MRI untuk memeriksa bagian dalam tubuh sangat efektif karena memiliki kemampuan membuat citra potongan *axial*, *sagittal* dan *coronal* tanpa banyak memanipulasi tubuh pasien serta diagnosa yang didapatkan akan lebih detail dan akurat. MRI merupakan peralatan radiologi terbaik untuk melakukan diagnosa tumor otak pada kasus yang rumit dan intensitasnya beragam (Balafar dkk, 2010).

Sebagian besar rumah sakit masih menggunakan MRI berkekuatan 1,5 tesla untuk pemeriksaan radiologi. Rumah Sakit X merupakan salah satu rumah sakit yang menyediakan jasa layanan pemeriksaan radiologi menggunakan MRI berkekuatan 1,5 tesla dan 3 tesla. MRI 3 tesla dapat melakukan semua fungsi MRI 1,5 tesla dengan hasil yang lebih baik. Hasil MRI 3 tesla memberikan resolusi ruang yang lebih baik dan sangat bermanfaat untuk menghasilkan pencitraan pembuluh darah (RS Premier Surabaya, 2017). Semakin baik kualitas hasil yang diperoleh maka semakin mahal juga biaya untuk melakukan pemeriksaan. Sehingga kebanyakan orang lebih memilih menggunakan MRI 1,5 tesla karena harganya lebih murah dari pada MRI 3 tesla. Berdasarkan ketersediaan dan kualitas citra yang dihasilkan oleh MRI 1,5 tesla dilakukan pendekatan untuk menentukan letak ketidaknormalan dalam kasus ini yaitu letak tumor otak.

Banyak peneliti mencari metode-metode yang dapat mempermudah dan mempercepat dalam pemerikasaan pada pasien. Dalam pengolahan citra MRI, beberapa metode dikembangkan untuk memperoleh segmentasi dari citra. Untuk klasifikasi, metode yang banyak digunakan diantaranya adalah Neural Network (NN).

Neural Network dipilih karena dinilai mampu mengadaptasi cara kerja neuron manusia. Terdapat pengembangan metode dalam NN seperti *Multilayer Perceptron* (MLP) yang sangat baik digunakan jika *Region of Interest* (ROI) berada di tengah citra, namun keakuratannya akan menurun saat ROI tidak berada di tengah citra. Pada kenyataannya, letak tumor yang menjadi ROI, bisa berada dimana saja. Beberapa metode segmentasi citra dalam statistika komputasi yang pernah dilakukan adalah dengan model *based clustering*. Yaitu penelitian oleh Islamiyah (2018), Safa (2018), Qunita (2018) dan Solichah (2018) dengan menggunakan *Finite mixture model*. Berbeda dengan penelitian sebelumnya yang menggunakan metode *clustering* sebagai dasar analisis. Pada penelitian ini menggunakan metode klasifikasi sebagai dasar analisis, yaitu dengan pendekatan metode *Fully Convolutional Network* (FCN). Penggunaan metode klasifikasi pada penelitian ini untuk memperoleh segmentasi dari tumor otak sehingga dapat langsung menentukan kelas tanpa memperkirakan jumlah claster terlebih dahulu. Pemilihan metode ini juga didasarkan atas rekomendasi penelitian Long (2018) yang menemukakan bahwa klasifikasi citra dapat digunakan dalam segmentasi citra dengan akurasi *pixel* 85,2%. Berdasarkan penjelasan diatas, pada tugas akhir ini akan dilakukan segmentasi citra MRI 1,5 tesla tumor otak dengan metode *Fully Convolutional Network* (FCN) agar dapat mengidentifikasi letak dari tumor dengan tepat. Diharapkan identifikasi tumor yang tepat akan dapat membantu dokter dan radiologis Rumah Sakit X dalam menentukan tindakan medis pada pasien tumor otak.

1.2 Rumusan Masalah

Menentukan seorang pasien menderita tumor otak perlu dilakukan pemeriksaan lebih lanjut yaitu salah satunya dengan mengambil gambar/citra MRI pada bagian dalam tubuh untuk memastikan terdapat tumor pada tubuh. Hasil citra yang diperoleh dari MRI bermacam-macam dan diperlukan beberapa gambar untuk menentukan daerah-daerah dari tumor otak, sehingga pada penelitian ini akan diselesaikan segmentasi tumor otak menggunakan *Fully Convolutional Network* (FCN). *Fully Convolutional*

Network merupakan metode *deep learning* yang bagus untuk pengklasifikasikan dan segmentasi citra, serta menentukan akurasi dari hasil segmentasi tumor otak.

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah mendapatkan hasil segmentasi tumor otak yang merupakan ROI dari citra MRI 1,5 tesla menggunakan *Fully Convolutional Network (FCN).* Memperoleh model asitektur terbaik dengan membandingkan jumlah *convolutional block* dan *convolutional layer* yang digunakan dalam pembentukan model. Memperoleh akurasi dari data testing dalam segmentasi tumor otak.

1.4 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah memperoleh hasil segmentasi tumor otak sehingga dapat diperoleh gambaran daerah tumor otak dengan akurat pada citra MRI 1,5 tesla. Dapat memberikan kontribusi bagi dunia penelitian terutama dalam bidang penentuan lokasi tumor otak.

1.5 Batasan Masalah

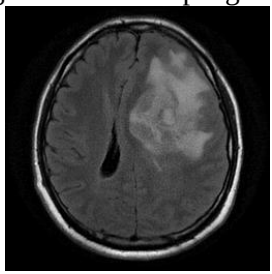
Batasan masalah yang digunakan dalam penelitian ini adalah data yang digunakan bersumber dari tiga pasien tumor otak yang berasal dari RS.X. Citra MRI yang digunakan berasal dari mesin MRI berkekuatan 1,5 tesla.

BAB II
TINJAUAN PUSTAKA

2.1 *Magnetic Resonance Imaging* (MRI)

Magnetic Resonance Imaging (MRI) adalah suatu alat kedokteran bidang pemeriksaan diagnostik radiologi, yang menghasilkan rekaman gambar potongan penampang tubuh/organ dalam manusia dengan menggunakan medan magnet berkekuatan antara 0,064 – 1,5 tesla (1 tesla = 1000 Gauss) (Stark, 1988). MRI digunakan untuk memperjelas gambaran bagian dalam manusia yang tidak tampak mata. Memperoleh gambaran secara jelas dari bagian dalam tubuh dapat mempercepat dan lebih akurat dalam pengambilan tindakan terhadap suatu penyakit. Keunggulan pencitraan medis MRI jika dibandingkan dengan pencitraan medis lainnya adalah sebagai berikut (Soesanti dkk., 2010) :

1. MRI unggul untuk mendeteksi beberapa kelainan pada jaringan lunak seperti otak (Gambar 2.1) dan sumsum tulang.
2. Mampu memberi gambaran detail anatomi dengan lebih jelas.
3. Mampu melakukan pemeriksaan fungsional yang lebih baik.
4. Mampu membuat gambaran potongan *axial*, *coronal*, dan *sagittal* (Gambar 2.2) tanpa mengubah posisi pasien. Masing-masing citra potongan ini dapat terdiri atas beberapa lapisan dengan ketebalan lapisan yang ditentukan.
5. MRI tidak menggunakan radiasi pengion.



Gambar 2.1 MRI tumor otak

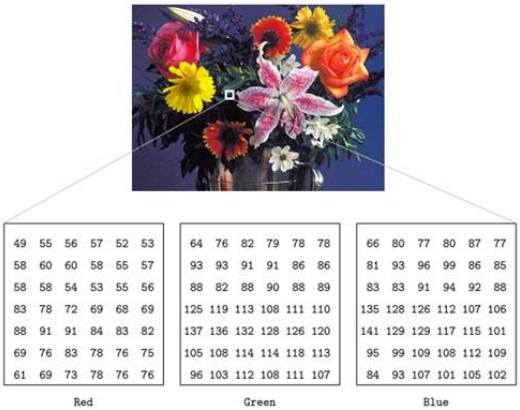
Pemeriksaan MRI secara umum menghasilkan 4 gambaran yaitu T1, T2, T2 *flair* dan T1-C. Dengan menggabungkan dari ke-

empat jenis gambar dan ketiga potongan bidang tersebut seorang dokter dapat mengetahui kelainan yang diderita oleh pasien.

Gambar 2.2 Citra MRI otak

2.2 Citra Digital

Citra adalah gambar diam (foto) maupun gambar bergerak (yang berasal dari *webcam*). Sedangkan digital disini mempunyai maksud bahwa pengolahan citra/gambar dilakukan secara digital menggunakan komputer (Sutoyo dkk, 2009). Citra digital secara umum dapat dibagi menjadi 3 yaitu *color image*, *black and white image* dan *binary image*.



Gambar 2.3 *Color Image*

Gambar RGB atau *color image* (Gambar 2.3) pada masing-masing *pixel* memiliki warna tertentu yang terdiri dari *Red*, *Green*, dan *Blue*. Masing-masing warna memiliki interval nilai 0 hingga

255 sehingga terdapat total 255^3 kombinasi warna yang dapat digunakan. Pada color image gambar terdiri dari tiga matriks yang dijadikan satu, setiap matriks mewakili nilai merah, biru, dan hijau untuk setiap *pixel* pada gambar (McAndrew, 2016).

Citra digital *black and white* (*grayscale*) terdiri dari gradasi warna dari hitam hingga putih untuk setiap *pixel* pada gambar. Rentang warna pada *black and white* sangat cocok digunakan dalam pengolahan file gambar (McAndrew, 2016). *Black and white* sebenarnya merupakan hasil penghitungan dari *color image*, dengan persamaan (2.1)

$$I_{BW}(x,y) = 0,299I_R(x,y) + 0,587I_G(x,y) + 0,114I_B(x,y) \quad (2.1)$$

keterangan :

$I_{BW}(x,y)$ = nilai *pixel black and white* pada (x,y),

$I_R(x,y)$ = nilai *pixel red* pada (x,y),

$I_G(x,y)$ = nilai *pixel green* pada (x,y),

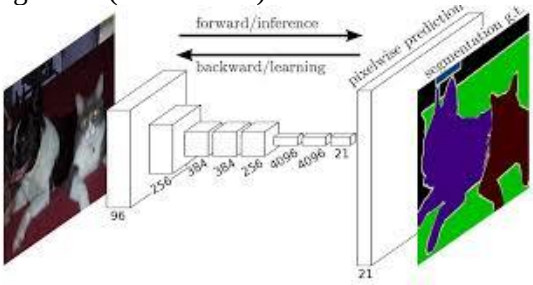
$I_B(x,y)$ = nilai *pixel blue* pada (x,y).

Terdapat banyak pengolahan gambar yang dapat digunakan, salah satunya adalah augmentasi gambar. Augmentasi merupakan metode memperbanyak data gambar untuk *training* dengan membuat perubahan untuk memperoleh gambar *training* yang serupa tetapi berbeda. Memotong gambar dengan cara yang berbeda dapat memunculkan posisi gambar yang berbeda pula sehingg mengurangi ketergantungan model pada posisi dimana objek muncul. Dapat juga dengan menyesuaikan kecerahan, warna, serta faktor-faktor lain untuk mengurangi sensitivitas model terhadap warna. Melakukan flip ke kiri dan ke kanan pada gambar biasanya tidak mengubah kategori dari objek. Metode ini adalah salah satu metode augmentasi gambar yang sering digunakan (Zhang dkk, 2019).

2.3 *Fully Convolutional Network*

Fully Convolutional Network (FCN) merupakan metode pengembangan *Convolution Neural Network* (CNN) yang dipergunakan dalam segmentasi gambar. CNN merupakan salah satu metode *deep learning* yang baik digunakan dalam klasifikasi citra.

Convolutional Neural Network terdiri dari *convolution*, *pooling* dan *fully connected* sebagai tiga *layer* utamanya, sedangkan pada FCN *fully connected layer* digantikan dengan *convolution layer*. Hal ini yang menyebabkan FCN dapat mengklasifikasikan setiap *pixels* pada gambar (Gambar 2.4).



Gambar 2.4 Arsitektur *Fully Convolutional Network*

FCN dibagi menjadi dua proses yaitu *Encoder* dan *Decoder*. Tahap e*ncoder* merupakan proses mengurangi ukuran dari matriks dengan memperkaya informasi dengan memperbanyak jumlah dari *feature maps* yang terbentuk. Sedangkan pada *decoder* merupakan proses pengembalian ukuran matriks ke ukuran semula serta memperkecil jumlah *feature maps* yang digunakan agar dapat dilakukan segmentasi dan membandingkan dengan target pada *ground truth* untuk masing-masing *pixel*. Menerapkan *convolution layer* serta *pooling layer* secara signifikan akan mengurangi jumlah parameter (pembagian berat) jaringan dan jaringan mempelajari korelasi antara *pixel*s disekitarnya (LeCun dan Bengio, 1995).

Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Operasi konvolusi merupakan operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi *output* sebagai *feature map* dari input citra (Gambar 2.5). *Convolutional layer* terdiri dari neuron yang tersusun sehingga membentuk sebuah filter dengan panjang dan tinggi (*pixel*). *Filter* ini akan digeser keseluruh bagian dari gambar. Setiap pergeseran akan dilakukan operasi "dot" antara input dan nilai dari filter tersebut sehingga menghasilkan sebuah *output* atau biasa disebut sebagai *activation map* atau *feature map*.

Berikut merupakan persamaan operasi konvolusi (2.2) (Naceur dkk, 2018).

$$Y_{i,j} = b + \sum_{k}\sum_{l} W_{k,l} \times X_{i+k-1,\, j+l-1} \qquad (2.2)$$

keterangan :

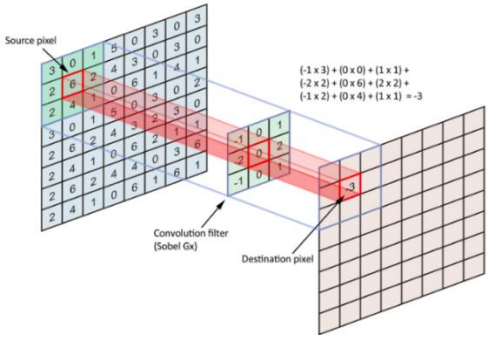
$Y_{i,j}$ = nilai *feature map* (*i*,*j*) ,

b = nilai bias,

$W_{k,l}$ = bobot ke-*k*, *l*,

$X_{i+k-1,\, j+l-1}$ = nilai *input*,

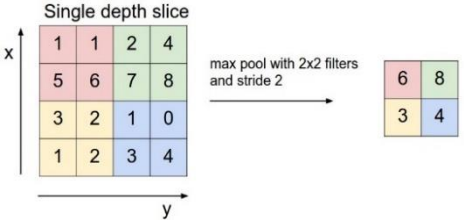
i = 1,2, ..., *I*,
j = 1,2, ..., *J*,
k = 1,2, ..., *K*,
l = 1,2, ..., *L*,



Gambar 2.5 *Convolution Layer*

Pooling merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling*. *Pooling layer* biasanya berada setelah *convolution layer*. Pada dasarnya *pooling layer* terdiri dari sebuah filter dengan ukuran dan *stride* tertentu yang akan secara bergantian bergeser pada seluruh area *feature map*. Penggunaan *pooling layer* pada FCN bertujuan untuk mereduksi ukuran citra sehingga dapat dengan mudah digantikan dengan sebuah layer

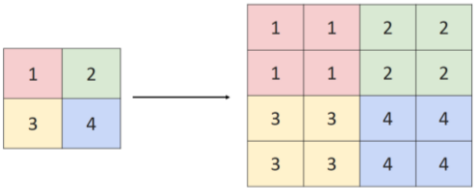
convolution dengan *stride* yang sama dengan *pooling layer* yang bersangkutan (Rawat dan Wang, 2017). *Stride* merupakan nilai dari banyak pergeseran dari layer yang digunakan baik *convolution layer* maupu *pooling layer*. *Padding "same"* merupakan penambahan angka 0 pada sekitar matrik gambar agar ukuran yang dihasilkan saat proses *convolution* tetap sama.



Gambar 2.6 *Max Pooling Layer*

Pooling layer yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur model FCN dapat secara progresif mengurangi ukuran dari *output* pada *feature map*. Hal ini dapat mengurangi jumlah parameter dan perhitungan pada matrik dan untuk mengendalikan *overfitting*. Terdapat beberapa jenis dari *pooling* yang dipergunakan dalam metode *deep learning* seperti *average pooling* dan *maximum pooling.* Banyak penelitian tentang citra yang menggunakan *maximum pooling* untuk mengambil nilai tertinggi dari *pixel* sesuai dari ukuran dan *stride* dari *pooling.* Gambar 2.6 menunjukkan contoh *maximum pooling* dengan input matrik berukuran 4×4 dengan satu *feature map* (sisi kiri) dan *max pooling* dengan ukuran 2×2 dengan *stride* 2 menghasilkan *output* matrik ukuran 2×2 (sisi kanan).

Dalam tahap *decoder* terdapat langkah dimana matriks diperbesar ukurannya dengan cara menggunakan *up-convolution*. Cara kerja dari *up-convolution* adalah dengan memperbanyak nilai dari suatu matrik yang kemudian akan diletakkan bersebelahan dengan asal pengambilan nilai. Setelah ukuran matrik diperbesar dilakukan pengambilan *feature maps* dari matrik dengan ukuran yang sama pada tahap *encoder* untuk memperkaya *feature maps*. Gambar 2.7 merupakan gambaran kerja dari *up-convolution.*



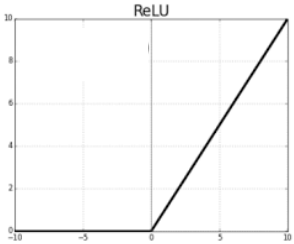
Gambar 2.7 *Up-Convolution Layer*

2.4 Fungsi Aktivasi ReLU

Fungsi aktivasi merupakan fungsi yang digunakan pada jaringan saraf untuk mengaktifkan atau tidak mengaktifkan neuron. ReLU (*Rectified Linear Unit*) merupakan fungsi aktivasi yang menghilangkan *vanishing gradient* dengan cara menerapkan fungsi aktivasi tersebut sebagai persamaan (2.3).

$$f(x) = \max(0, x) \tag{2.3}$$

Maka fungsi aktivasi ini akan memaksa nilai negatif untuk menjadi 0, menurut Krizhevsky (2012). Gambar 2.7 merupakan grafik fungsi aktivasi ReLU.



Gambar 2.7 Fungsi Aktvasi ReLU

2.5 Fungsi Aktivasi *Sigmoid*

Sigmoid merupakan fungsi aktivasi yang memiliki nilai pada *range* 0 sampai 1. Oleh karena itu, fungsi ini sering digunakan untuk nilai *output* yang terletak pada interval 0 sampai dengan 1 (Zhang dkk, 2019). Namun, fungsi ini bisa juga digunakan oleh jaringan saraf yang nilai *output* untuk pengklasifikasian dengan kode 0 atau 1. Berikut merupakan persamaan fungsi aktivasi *sigmoid* (2.4).

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.4}$$

2.6 *Loss Function*

Loss adalah nilai dari *error* antara nilai hasil prediksi dan nilai sebenarnya. Pada *machine learning* digunakan *loss function* digunakan untuk menghitung nilai dari *error* (Zhang dkk, 2019). Terdapat banyak fungsi yang dapat digunakan dalam menghitung *error*. Pada kasus penghitungan *loss* pada kasus yang memiliki dua kelas dapat digunakan *binary cross-entropy* dengan persamaan (2.5).

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \times \log(p(y_i) + (1-y_i) \times \log(1-p(y_i)) \tag{2.5}$$

keterangan :

N = jumlah banyak data,

y_i = kelas pada klasifikasi (0 atau 1),

$p(y_i)$ = nilai peluang y_i.

2.7 Akurasi

Akurasi merupakan kedekatan antara nilai prediksi dengan nilai sebenarnya. Dalam menentukan ketepatan akurasi dari pengujian digunakan *confusion* matrik. (Han dkk, 2013)

Tabel 2.1 *Confusion* Matrik

Nilai Sebenarnya	Nilai Prediksi	
	True	*False*
True	*TP*	*FN*
False	*FP*	*TN*

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

keterangan *:*

TP = True Positif

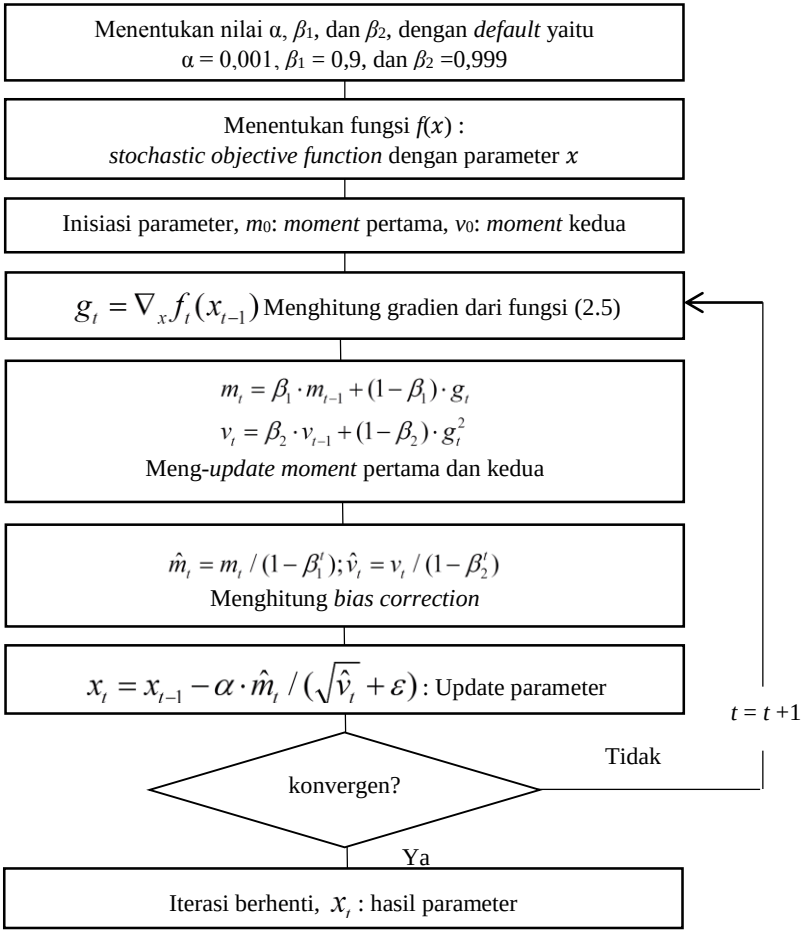
FP = False Positif

FN = False Negatif

TN = True Negatif

2.8 Optimasi Adam

Adam adalah metode tingkat pembelajaran adaptif. Adam dapat dilihat sebagai kombinasi RMSprop *dan Stochastic Gradient Descent* dengan momentum (Zhang dkk, 2019). Gambar 2.8 merupakan diagram alir untuk algoritma Adam.



Gambar 2.8 Diagram Alir Algoritma Adam

Optimasi digunankan untuk memperbarui dari nilai bobot parameter dan meminimalkan dari nilai *loss* pada saat dilakukan *training*. Pada metode FCN digunakan *convolution layer* pada pengaplikasian paramater, sehingga berikut merupakan persamaan yang digunakan.

$$g_t = \frac{\partial E}{\partial x},$$

Persamaan dari E adalah persamaan *loss function* (2.5)

$$g_t = \frac{\partial\left(-\frac{1}{N}\sum_{i=1}^{N} y_i \log(p(y_i)) + (1-y_i)\log(1-p(y_i))\right)}{\partial x},$$

Persamaan dari $p(y_i)$ adalah persamaan *activation function sigmoid* (2.4)

$$g_t = \frac{\partial\left(-\frac{1}{N}\sum_{i=1}^{N} y_i \log(\frac{1}{1+e^{-x}}) + (1-y_i)\log(1-\frac{1}{1+e^{-x}})\right)}{\partial x},$$

$$g_t = \partial(-\frac{1}{N}\sum_{i=1}^{N} y_i\left(\log(1)-\log(1+e^{-x})\right)+$$

$$(1-y_i)\left(\log(-e^{-x})-\log(1+e^{-x})\right))\frac{1}{\partial x},$$

$$g_t = -\frac{1}{N}\sum_{i=1}^{N} y_i\left(-\frac{1}{(1+e^{-x})\ln 10}\right)+$$

$$(1-y_i)\left(\frac{1}{-e^{-x}\ln(10)}-\frac{1}{(1+e^{-x})\ln(10)}\right) \qquad (2.7)$$

keterangan :

g_t = gradien fungsi

N = jumlah banyak data,

x = parameter

y_i = kelas pada klasifikasi (0 atau 1),

$p(y_i)$ = nilai peluang y_i.

2.9 *Correct Classification Ratio* (CCR)

Dalam mengecek kebaikan hasil segmentasi yang dihasilkan terhadap *ground truth* pada masing-masing citra MRI digunakan CCR sebagai akurasi. Berikut merupakan persamaan CCR (2.8).

$$\text{CCR} = \sum_{j=1}^{2} \frac{\left| GT_j \cap Seg_j \right|}{\left| GT \right|} \tag{2.8}$$

keterangan :

GT_j = *ground truth* pada *j*,

j bernilai 1 menunjukkan daerah non-ROI, dan *j* bernilai 2 menunjukkan daerah ROI tumor otak,

Seg_j = *pixel* hasil segmentasi.

2.10 *Ground truth*

Ground truth merupakan target yang digunakan dalam menentukan kelompok atau kelas pada gambar. Pada gambar *Ground truth* memiliki nilai tertentu untuk setiap dari *pixel* sesuai dengan kelompok atau kelas yang diinginkan (Gambar 2.9).

Gambar 2.9 *Ground truth*

Ground truth digunakan dalam menentukan kelompok atau kelas pada gambar sehingga dalam satu gambar dapat diketahui kelompok atau kelas yang ingin digunakan sebagai target dalam klasifikasi. *Ground truth* dibuat oleh seorang ahli atau orang yang dianggap dapat membedakan target dalam gambar sehingga dapat

digunakan dalam penentuan target dalam klasifikasi. Pada umunya warna di dalam gambar *ground truth* adalah hitam putih, dimana putih merupakan daerah ROI sedangkan warna hitam merupakan daerah non ROI.

BAB III
METODOLOGI PENELITIAN

3.1 Sumber Data

Data yang digunakan pada penelitian ini merupakan data sekunder yaitu citra MRI pada penderita tumor otak. Citra MRI yang digunakan berasal dari rekam medis tiga pasien di RS. X yang diambil dari tahun 2017 dengan *scan* mesin MRI berkekuatan 1,5 Tesla. Dilakukan pengambilan 76 *slice* citra MRI T1-C dan T2 *flair* pada bagian *axial* yang memiliki tumor otak dengan rincian pada Tabel 3.1.

Tabel 3.1 Pembagian data

Pasien	Jumlah *slice*	
	T1-C	T2 *Flair*
1	11	15
2	13	15
3	7	9

Citra berukuran 256×256 *pixels* pada bagian *axial* dengan format dicom (.dcm). Data *Ground truth* didapatkan dari dokter di RS.X, yang telah melakukan tindakan lebih lanjut terhadap pasien tumor otak. Setiap satu citra MRI tumor otak memiliki satu *ground truth* yang khusus dan tidak dapat ditukar-tukar sebagai target klasifikasi untuk setiap *pixels* pada citra. Sehingga terdapat 76 citra *ground truth* dengan ukuran 256×256 *pixels* dengan format JPEG (.jpg).

3.2 Variabel Penelitian

Variabel penelitian yang digunakan pada penelitian ini dapat dilihat pada Tabel 3.2.

Tabel 3.2 Variabel Penelitian

Variabel	Keterangan
$X_{1,i,j}$	Nilai *Red* setiap *pixel*
$X_{2,i,j}$	Nilai *Green* setiap *pixel*
$X_{3,i,j}$	Nilai *Blue* setiap *pixel*
$\mathbf{Y}_j$	*Matrix Ground truth*

dengan :

$0 < i \leq K$; K bernilai 65.536 dari total *pixel* pada setiap gambar (256 × 256 = 65.536).

$0 < j \leq L$; L bernilai sejumlah citra MRI yang digunakan.

Y *matrix* berukuran 256 × 256.

3.3 Struktur Data

Struktur data yang akan digunakan dalam penelitian ini terbagi dalam struktur data untuk citra MRI dan *ground truth.* Struktur data untuk citra MRI ditunjukkan pada Tabel 3.3 dan Tabel 3.4 untuk struktur data *ground truth*.

Tabel 3.3 Struktur Data MRI

No	$X_{1,1}$	$X_{1,2}$	…	$X_{1,65536}$	$X_{2,1}$	$X_{2,2}$	…	$X_{2,65536}$	…	$X_{3,65536}$
1	$X_{1,1,1}$	$X_{1,2,1}$	…	$X_{1,65536,1}$	$X_{2,1,1}$	$X_{2,2,1}$	…	$X_{2,65536,1}$	…	$X_{3,65536,1}$
2	$X_{1,1,2}$	$X_{1,2,2}$	…	$X_{1,65536,2}$	$X_{2,1,2}$	$X_{2,2,2}$	…	$X_{2,65536,2}$	…	$X_{3,65536,2}$
…	…	…	…	…	…	…	…	…	…	…
L	$X_{1,1,l}$	$X_{1,2,l}$	…	$X_{1,65536,l}$	$X_{2,1,l}$	$X_{2,2,l}$	…	$X_{2,65536,l}$	…	$X_{3,65536,l}$

Tabel 3.4 Struktur Data *Ground Truth*

No	Y_1	Y_2	…	Y_{65536}
1	$Y_{1,1}$	$Y_{2,1}$	…	$Y_{65536,1}$
2	$Y_{1,2}$	$Y_{2,2}$	…	$Y_{65536,2}$
…	…	…	…	…
L	$Y_{1,l}$	$Y_{2,l}$	…	$Y_{65536,l}$

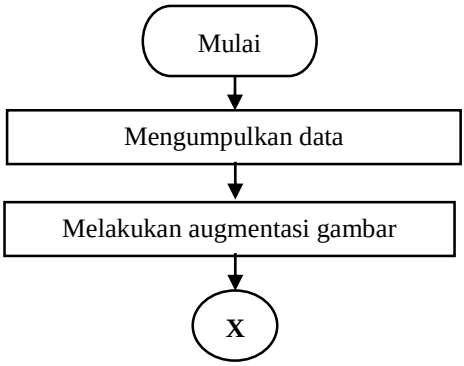
3.4 Langkah Analisis

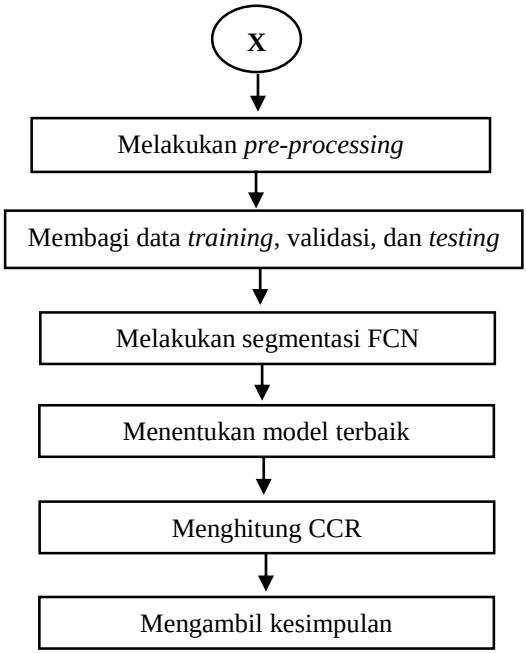
Langkah analisis yang dilakukan pada penelitian ini adalah sebagai berikut:

1. Mengumpulkan data citra MRI otak dan *ground truth*.
2. Melakukan augmentasi citra MRI dan *ground truth* dengan cara *flip horizontal*.
3. Melakukan *pre-processing* data sebagai berikut.
   a. Merubah citra MRI menjadi matrik angka berdasarkan sub bab 2.2.
   b. Merubah *ground truth* menjadi *grayscale* dengan persamaan 2.1.

   c. Menormalisasi nilai setiap *pixel*.
4. Membagi data *training*, validasi, dan *testing*.
5. Melakukan segmentasi dengan metode *Fully Convolutional Network* berdasarkan sub bab 2.3.
   a. Tahapan *encoder* dengan mengurangi ukuran dari matrik citra serta memperbanyak *feature maps*.
   b. Tahapan *dencoder* dengan mengembalikan ukuran matrik citra ke ukuran semula untuk dibandingkan dengan *grount truth* (target) serta memperkecil *feature maps* yang dipergunakan.
   c. Melakukan klasifikasi pada setiap *pixel* dengan persamaan 2.4.
   d. Meng-*update* parameter dengan *backpropagation* dengan persamaan 2.7
   e. Menghitung nilai *loss* dan akurasi untuk setiap model
6. Menentukan model terbaik berdasarkan nilai *loss* dan akurasi terakhir pada setiap model.
7. Menghitung nilai *Correct Classification Ratio* menggunakan model terbaik dengan data *testing* berdasarkan sub bab 2.7.
8. Mengambil kesimpulan.

Berdasarkan langkah-langkah analisis yang telah dijelaskan dapat dibuat diagram alir penelitian untuk analisis FCN pada Gambar 3.1.





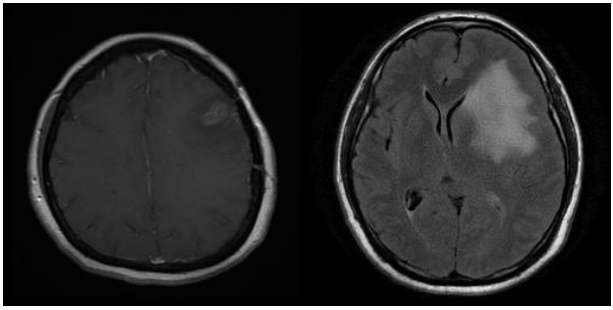
Gambar 3.1 Diagram Alir Langkah Analisis

BAB IV
ANALISIS DAN PEMBAHASAN

Pada bab ini dilakukan klasifikasi untuk setiap *pixel* pada citra MRI dengan target *ground truth* yang merupakan ROI tumor otak. Pengklasifikasian *pixel* pada citra MRI menggunakan *Deep learning* dengan metode *Fully Convolutional Network* untuk memperoleh hasil segmentasi citra MRI tumor otak. Digunakan data *training* dan data validasi untuk memperoleh model terbaik dan akan diujikan dengan data *testing*. Kemudian dilakukan segmentasi tumor otak untuk memperoleh gambaran tumor pada citra MRI.

4.1 Karakteristik Citra MRI Tumor Otak

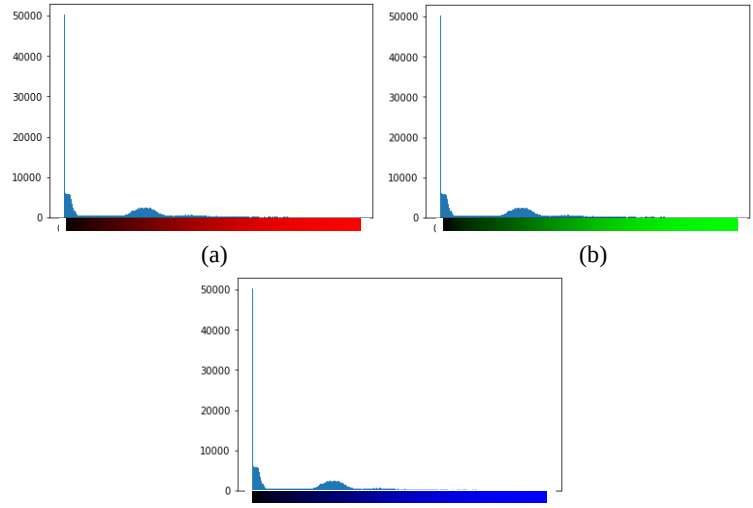
Dari citra MRI yang digunakan pada penilitian ini yaitu T2 *flair* dan T1-C terdapat perbedaan dari karakteristik dari ROI. Pada ROI citra MRI T2 *flair* ditandai dengan kecenderungan berwarna putih. Sedangkan pada T1-C, ROI ditandai dengan warna lebih ke abu-abuan dan warna lebih gelap pada sekitar warna abu-abu.



Gambar 4.1 Citra MRI T1-C (kiri) dan T2 *Flair* (kanan)

Gambar 4.1 merupakan merupakan contoh gambar dari tumor otak pada salah satu *slice* pada citra MRI penderita tumor otak. Citra MRI diperoleh dari melihat persebaran nilai dari *pixel* pada citra dengan nilai 0 hingga 255. Histogram dari citra menunjukkan pola dari intensitas dari warna sehingga didapat pola persebaran nilai *pixel* warna pada citra MRI. Pada citra diperoleh

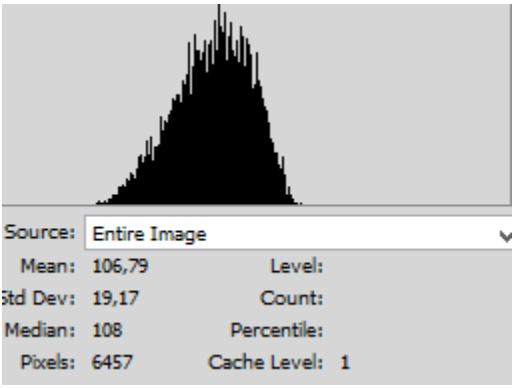
tiga histogram untuk masing-masing warna pembentuk yaitu warna merah, hijau dan biru.



Gambar 4.2 Histogram *pixels* citra MRI (a) Merah, (b) Hijau, dan (c) Biru

Gambar 4.2 merupakan histogram dari salah satu dari *slice* dari bagian *axial* pada T2 *Flair* yaitu *slice* ke-18 pada pasien ke-1. Hasil seluruh histogram memiliki nilai persebaran yang sama, baik histogram warna merah, hijau dan biru sehingga pada histogram RGB juga memiliki nilai yang sama. Histogram RGB memiliki nilai *mean* 43,31, dengan median 40 dan standar deviasi 45,19 yang berasal dari 65.536 *pixel*s. Pada ketiga histogram didominasi oleh nilai 0, hal ini terjadi karena gabungan dari nilai 0 untuk masing-masing histogram merah, hijau dan biru mengasilkan warna hitam yang merupakan background dari gambar.

Gambar 4.3 merupakan histogram dengan melihat daerah *Region of Interes* yang merupakan tumor otak dengan acuan dari *Ground truth*. Pada histogram RGB terlihat bahwa rentang warna terdapat pada rentang 49 hingga 151 dengan *mean* 106,79, standar deviasi 19,17 dan median 108 yang terletak pada 6.457 *pixel*s.

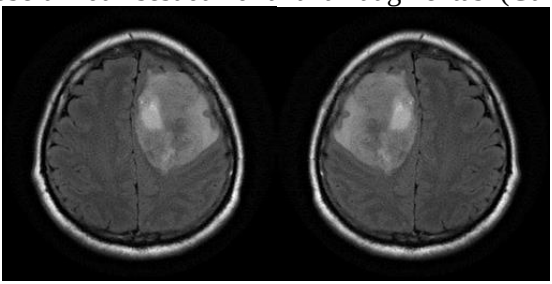


Gambar 4.3 Histogram *pixels* citra pada ROI dari *Ground truth*

Sebesar 9,85% merupakan tumor dari keseluruhan citra MRI (termasuk *background*). Tidak semua nilai warna RGB pada *pixel* rentang tersebut merupakan tumor, tetapi juga merupakan jaringan sehat dengan warna yang sama.

4.2 *Pre-processing* Data

Sebelum dilakukan analisis, dilakukan *pre-processing* data terlebih dahulu. Dilakukan perbanyakan data dengan cara meng-augmentasi citra MRI dan *ground truth* dengan cara *flip* secara *horizontal* sehingga jumlah citra yang sebanyak 152 untuk masing-masing citra MRI serta *ground truth*. Berikut merupakan contoh gambar sebelum dan sesudah dilakukan augmentasi (Gambar 4.4).

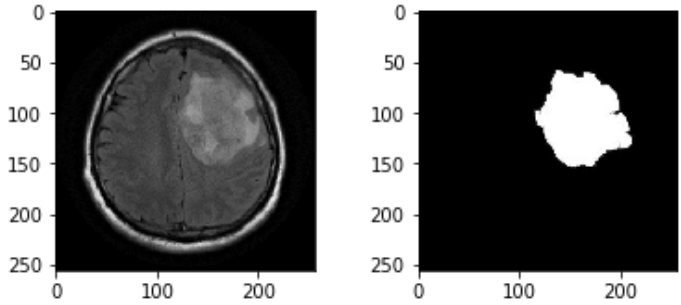


Gambar 4.4 Citra MRI sebelum (kiri) dan setelah (kanan) Augmentasi

Agar citra MRI dan gambar *ground truth* dapat diolah meng-gunakan metode FCN perlu dilakukan pengubahan citra menjadi

angka dalam bentuk matrik. Matrik yang terbentuk akan berukuran 3 dimensi yaitu $256 \times 256 \times 3$ dengan 256 yang pertama merupakan lebar *pixel* pada gambar, sedangkan 256 kedua merupaka tinggi *pixel* pada gambar dan 3 adalah lapisan RGB pada gambar (*feature maps*). Setiap *pixel* pada gambar memiliki warna yang spesifik sehingga akan tergantikan dengan nilai antara 0 hingga 255. Nilai pada matrik akan terisi dengan nilai 0 yang berarti warna hitam sedangkan semakin besar nilai akan mengikuti nilai dari lapisan warna. Sebagai contoh lapisan *Red* (merah) akan memiliki nilai 255 untuk warna merah terang.

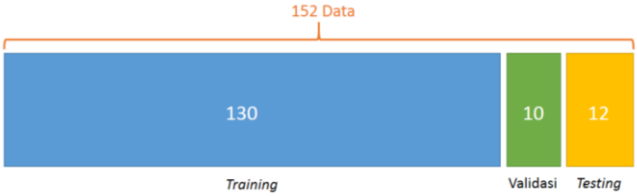
Setelah gambar telah menjadi matrik maka dilakukan perubahan gambar *ground truth* dari RGB menjadi *grayscale*. Pengubahan ini bertujuan agar dimensi dari matrik menjadi $256 \times 256 \times 1$. Setelah gambar *ground truth* menjadi *grayscale*, maka setiap *pixel* akan hanya memiliki satu nilai yang dipergunakan sebagai target ROI. Selanjutnya dilakukan normalisasi nilai pada masing-masing *pixel* pada gambar. Normalisasi dilakukan dengan membagi nilai pada setiap *pixel* dengan nilai tertinggi pada interval warna yaitu 255. Sehingga diperoleh nilai pada setiap *pixel* berada pada interval 0 sampai 1. Berikut merupakan contoh hasil dari *preprocessing* citra MRI dan *ground truth* seperti pada Gambar 4.5.



Gambar 4.5 Citra MRI otak (kiri) dan *Ground truth* (kanan)

Gambar 4.5 merupakan citra MRI dan ROI tumor otak pada *Ground truth* dari bagian *axial* T2 *flair* pada *slice* ke-18 dari pasien ke-1. Pengklasifikasian menggunakan *Ground truth* sebagai target

yaitu warna putih sebagai ROI tumor otak sedangkan warna hitam untuk jaringan non tumor otak. Dilakukan pembagian 3 kelompok data citra MRI hasil dari augmentasi dan *pre-processing* yaitu data *training*, validasi dan *testing* (Gambar 4.6).



Gambar 4.6 Pembagian data *training*, validasi dan *testing*

Dari seluruh data citra MRI dilakukan pengambilan 130 citra sebagai data *training*, 10 data validasi, dan 12 data *testing*. Data *testing* diambil dari satu citra T2 *Flair* dan T1-C untuk masing-masing pasien pada *slice* yang sama, serta hasil augmentasinya. Sehingga diperoleh 12 citra MRI sebagai data testing. Data validasi adalah 10 citra MRI yang diambil dari citra yang mewakili citra T2 *Flair* maupun T1-C. Sedangkan data *training* merupakan data 130 citra MRI sisa pembagian dari data validasi dan data *testing*. Data *training* digunakan untuk *update* paramater serta menangkap pola-pola yang dimiliki oleh citra MRI sehingga diperoleh model. Data validasi digunakan sebagai pembanding hasil akurasi dan *loss* dari data *training* antar model sehingga diperoleh model terbaik. Data *testing* dipergunakan untuk memperoleh CCR dari data yang baru.

4.3 Klasifikasi Tumor otak berdasarkan FCN

Fully Convolutional Network atau FCN merupakan metode segmentasi dengan cara mengklasifikasikan masing-masing *pixel* pada gambar. *Pixel* hasil klasifikasi akan membentuk segmentasi saat dikembalikan ke dalam bentuk gambar. Pada klasifikasi setiap *pixel* gambar disesuaikan dengan ROI sehingga hasil citra dibagi menjadi dua kelas yaitu tumor otak dan non-tumor otak.

Pada proses *training* digunakan *batch size* sebesar 13 dan jumlah *epoch* 150. *Batch size* dengan nilai 13 menyebabkan terambilnya 13 gambar sebagai sampel dan *update weight* dan *bias*,

sehingga untuk sekali *epoch* terjadi *update weight* dan *bias* sebanyak 10 kali. *Epoch* atau dapat disebut iterasi yang terdiri dari *update weight* dan *bias* berdasarkan *batch size* dan pengecekan rata-rata *loss* dan *akurasi* untuk data *training* dan data validasi. Total banyak *update weight* dan *bias* yang dilakukan untuk memperoleh satu model adalah sebanyak 1500 kali.

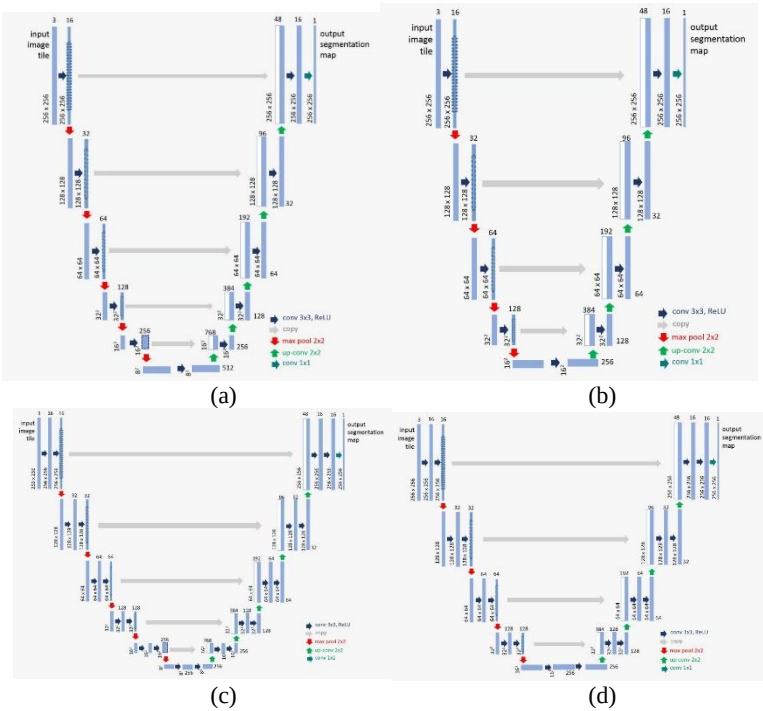
Jumlah *Convolution layer* dan *Convolution block* sangat berpengaruh pada pembentukan model terbaik karena akan mempengaruhi jumlah dari parameter yang digunakan dan *feature map* yang terbentuk. Karena itu, pada penelitian ini dilakukan *training* dengan membandingkan jumlah *convolution layer* yang digunakan dalam satu *block* dan jumlah *block* yang digunakan. Pada Tabel 4.1 terdapat rincian jumlah *convolution layer* untuk setiap *block* dan jumlah *convolution block* pada setiap model.

Tabel 4.1 Model *Training* yang Digunakan

Model	**Jumlah *Conv layer***	**Jumlah *Conv block***	**Jumlah parameter**
Model 1	1 Conv @ *block*	10 *block*	3.930.273
Model 2	1 Conv @ *block*	8 *block*	980.385
Model 3	2 Conv @ *block*	10 *block*	7.862.401
Model 4	2 Conv @ *block*	8 *block*	1.962.625

Perbedaan arsitektur dari model 1, 2, 3 dan 4 dapat terlihat pada Gambar 4.4. Gambar 4.4 (a) merupakan gambar arsitektur untuk model 1, Gambar 4.4 (b), (c) dan (d) secara berturut-turut adalah arsitektur untuk model 2, 3 dan 4. Dari model 1 dan 2 memiliki satu *convolution layer* untuk masing-masing *convolution block* menyebabkan berkurangnya jumlah parameter lebih dari setengah dari model yang memiliki dua *convolution layer* untuk setip *convolution block* yaitu model 3 dan 4. Pada model 1 dan 3 digunakan 5 *convolution block* pada tahap *encoder*. Pada setiap output tahap *convolution block* menghasilkan matriks dengan ukuran setengah dari ukuran semula sehingga diperoleh matrik ukuran 1/32 pada akhir tahapan *encoder*. Model 2 dan 4 memiliki 4 *convolution block* pada tahap *encoder* yang menyebabkan ukuran akhir matrik 1/16 dari ukuran semula. Model 1 dan 3 memiliki

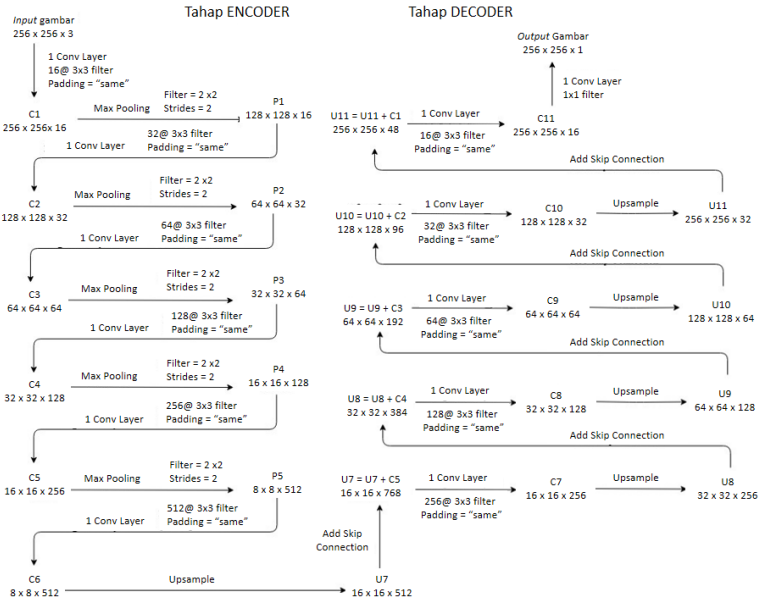
ukuran matrik pada akhir tahap *encoder* adalah 8×8 sedangkan pada model 2 dan 4 memiliki ukuran akhir 16×16.



Gambar 4.7 Arsitektur model 1 (a), model 2 (b), model 3 (c), dan model 4 (d).

Berikut merupakan ilustrasi dari model 1 yang terdiri dari 5 *convolution block* untuk *encoder* dan 5 *convolution block* pada *decoder* serta menggunakan satu *convolution layer* untuk setiap *convolution block* (Gambar 4.8). Pada *convolution block* bagian *encoder* terdiri dari *convolution layer* lalu dilakukan aktivasi fungsi ReLU dan dilanjutkan dengan *max pooling layer*. C1, C2, ... ,C11 merupakan matrik hasil *output* dari proses *convolution layer*. *Convolution layers* yang digunakan ukuran 3×3 dengan *padding "same"* dan *stride* 1. Nilai *stride* 1 menunjukkan pergeseran dari filter pada operasi *convolutional layer* bergeser satu *pixel* untuk setiap kali operasi. Penggunaan *padding "same"* digunakan untuk

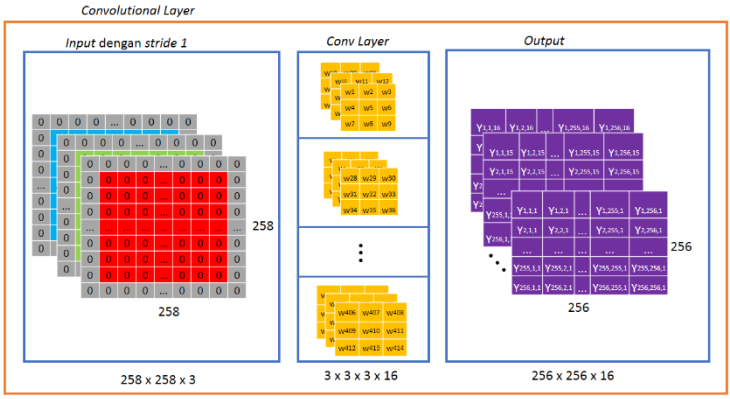
mempertahankan ukuran dari matrik citra MRI setelah melalui *convolution layer*.



Gambar 4.8 Ilustrasi model 1.

Berikut merupakan ilustrasi operasi *convolution layer* pada C1 (Gambar 4.9). *Input* merupakan citra MRI dengan ukuran 256×256 dengan *feature maps* 3 yaitu lapisan merah, hijau dan biru atau dapat dituliskan $256 \times 256 \times 3$. Penggunaan *padding "same"* dengan menambahkan vektor nilai 0 pada sekitar matrik. Pada penelitian ini digunakan *convolution layer* dengan ukuran 3 $\times 3$ dan *stride* 1 maka terjadi penambahan satu lapis nilai vektor 0 pada sekitar matrik. Ukuran matrik setelah disesuaikan dengan *padding "same"* menjadi $258 \times 258 \times 3$. Pada *convolution layer* memiliki ukuran matrik 3×3 serta dengan menyesuaikan *feature maps* dari *input* maka untuk menghasilkan satu *feature map* yang baru ukuran *filter* matriks akan menjadi $3 \times 3 \times 3$. Pada C1 ingin dihasilkan 16 *feature maps* baru dari hasil operasi konvolusi sehingga diperlukan 16 *filter* matrik berukuran $3 \times 3 \times 3$. Dimana

untuk setiap operasi pada *convolution layer* yang dilakukan memiliki nilai bobot yang berbeda-beda.



Gambar 4.9 Ilustrasi *convolution layer* C1.

Pada *Output* C1 didapatkan 16 *feature maps* dengan ukuran matrik yang sama seperti diawal yaitu 256×256. Didapatkan 16 pola gambar baru dengan *input* citra MRI RGB. Berikut contoh persamaan untuk *convolution layer* pada *convolution block* yang pertama pada model 1.

$$Y_{1,1,1} = -0,01304X_{1,1,1} + 0,08496X_{1,2,1} + 0,05443X_{1,3,1} + 0,16343X_{2,1,1} + 0,16381X_{2,2,1} + 0,15175X_{2,3,1} + \ldots - 0,16235X_{3,3,3}$$

$$Y_{1,2,1} = -0,01304X_{1,2,1} + 0,08496X_{1,3,1} + 0,05443X_{1,4,1} + 0,16343X_{2,2,1} + 0,16381X_{2,3,1} + 0,15175X_{2,4,1} + \ldots - 0,16235X_{3,4,3}$$

$$\vdots$$

$$Y_{256,256,1} = -0,01304X_{256,256,1} + 0,08496X_{256,257,1} + 0,05443X_{256,258,1} + 0,16343X_{257,256,1} + 0,16381X_{257,257,1} + 0,15175X_{257,258,1} + \ldots - 0,16235X_{258,258,3}$$

$$\vdots$$

$$Y_{1,1,2} = -0,04959X_{1,1,1} + 0,06726X_{1,2,1} - 0,0078X_{1,3,1} +$$
$$-0,06495X_{2,1,1} - 0,17948X_{2,2,1} + 0,0745X_{2,3,1} + \ldots$$
$$-0,15989X_{3,3,3}$$

$$\vdots$$

$$Y_{256,256,16} = 0,03779X_{256,256,1} + 0,12983X_{256,257,1} +$$
$$-0,01291X_{256,258,1} + 0,00681X_{257,256,1} +$$
$$-0,06969X_{257,257,1} + 0,02662X_{257,258,1} + \ldots$$
$$-0,18755X_{258,258,3}$$

P1, P2, ..., P5 pada Gambar 4.8 adalah hasil *output* dari *max pooling* pada tahap *encoder*. *Max pooling* yang digunakan berukuran 2×2 dan *stride* sebesar 2 untuk masing-masing *feature maps*. Didapatkan 16 *feature maps* (matrik) baru dengan setiap *feature maps* berukuran setengah dari ukuran semula yaitu 128×128. Berikut merupakan persamaan saat dilakukan *max pooling* pada P1.

$$Y_{1,1,1} = \max(0, X_{1,1,1}, X_{1,2,1}, X_{2,1,1}, X_{2,2,1})$$
$$Y_{1,2,1} = \max(0, X_{1,3,1}, X_{1,4,1}, X_{2,3,1}, X_{2,4,1})$$

$$\vdots$$

$$Y_{2,1,1} = \max(0, X_{3,1,1}, X_{3,2,1}, X_{4,1,1}, X_{4,2,1})$$
$$Y_{2,2,1} = \max(0, X_{3,3,1}, X_{3,4,1}, X_{4,3,1}, X_{4,4,1})$$

$$\vdots$$

$$Y_{128,128,1} = \max(0, X_{255,255,1}, X_{255,256,1}, X_{256,255,1}, X_{256,256,1})$$
$$Y_{1,2,2} = \max(0, X_{1,3,2}, X_{1,4,2}, X_{2,3,2}, X_{2,4,2})$$

$$\vdots$$

$$Y_{128,128,16} = \max(0, X_{255,255,16}, X_{255,256,16}, X_{256,255,16}, X_{256,256,16})$$

Diperolehlah *output* matrik ukuran $128 \times 128 \times 16$ untuk satu kali *convolution layer* dan satu kali *max pooling* (satu *convolution block*). Selanjutnya *Output* dari *convolution block* yang pertama di

gunakan sebagai *input* pada *convolution block* kedua, dan begitu seterusnya hingga pada akhir dari proses tahapan *encoder*. Pada model 1 bagian *encoder* terdiri dari 5 *convolution block* (C1-C5, P1-P5), menyebabkan penurunan ukuran dari matrik yang melalui *pooling layer* hingga 1/32 ukuran original yaitu 8×8. Pengurangan dari ukuran matrik ini diimbangi dengan penambahan jumlah dari *feature maps* yang dibuat untuk menyimpan informasi-informasi pada masing-masing *block* yang kemudian akan dilakukan proses lebih lanjut. Tabel 4.2 menunjukkan jumlah *feature maps input* dan *output* serta jumlah dari parameter yang digunakan pada tahapan *encoder*.

Tabel 4.2 *Feature maps encoder*

	Jumlah *Feature Maps*		Jumlah
	Input	***Output***	Parameter
Input	-	3	-
C1	3	16	448
C2	16	32	4640
C3	32	64	18496
C4	64	128	73856
C5	128	256	295168

Jumlah *feature maps* yang terbentuk pada masing-masing tahapan ditentukan sebelumnya dengan semakin banyak proses *convolution* dan pengurangan ukuran dari matrik maka akan semakin banyak pula *feature maps* yang digunakan untuk menangkap pola dari gambar. Jumlah dari parameter akan bertambah sebanding dengan ukuran matrik *convolutional filter* dan jumlah *feature maps* yang ingin terbentuk.

Pada proses selanjutnya digunakan *convolution layer* (C6) tanpa melakukan *pooling* untuk memasuki tahapan *decoder*. *Input* untuk proses *decoder* pada model 1 adalah matrik berukuran 8×8 dengan jumlah *feature maps* 512 atau dapat ditulis $8 \times 8 \times 512$. Dilakukan proses *up-sampling* menggunakan *up-convolutional* untuk menggandakan ukuran matrik dengan menyalin nilai matrik dari sekitarnya. U7, U8, ... , U11 pada Gambar 4.8 merupakan operasi *up-convolutional* dari model 1. U7 merupakan proses *up-sampling* pertama dimana ukuran matrik input $8 \times 8 \times 512$ menjadi matrik

berukuran $16 \times 16 \times 512$. Setelah itu dilakukan penggabungan dari *feature maps* pada proses *encoder* yang memiliki ukuran matrik yang sama untuk dilakukan penambahan jumlah *feature maps* agar tidak kehilangan pola atau informasi dari proses sebelumnya. Selanjutnya dilakukan tahapan *convolution* dengan ukuran matrik pada *input feature maps* pada tahap *convolution layer* sebesar $16 \times 16 \times 768$ dan menghasilkan *output* matrik ukuran $16 \times 16 \times 256$ (C7). Setelah melalui proses *convolution* dilakukan *up-sampling* lagi, begitu seterusnya hingga ukuran matrik kembali ke ukuran semula yaitu 256×256.

Tabel 4.3 *Feature maps decoder*

	Jumlah *Feature Maps*		**Jumlah**
	input	***output***	**Parameter**
C7	768	256	1769728
C8	384	128	442496
C9	192	64	110656
C10	96	32	27680
C11	48	16	6928
Output	16	1	17

Pada Tabel 4.3 menunjukkan jumlah parameter dan *feature maps* pada proses *decoder* sehingga diperoleh pada *output* yaitu matrik berukuran $256 \times 256 \times 1$. Input pada C7-C11 merupakan gabungan anatra *feature maps* sebelumnya dan *feature maps* pada tahap *encoder*. Matrik *output* terakhir akan dibandingkan *ground truth* sehingga diperoleh akurasi dan *loss* dari hasil klasifikasi pada setiap *pixel*. Nilai dari masing-masing *pixel* dimasukkan kedalam fungsi aktivasi Sigmoid untuk memperoleh hasil klasifikasi. Berikut merupakan persamaan klasifikasi pada *output* gambar citra MRI tumor otak model 1.

$$Y_{1,1} = \frac{1}{1+e^{-X_{1,1}}}$$

$$Y_{1,2} = \frac{1}{1+e^{-X_{1,2}}}$$

$$\vdots$$

$$Y_{1,256} = \frac{1}{1+e^{-X_{1,256}}}$$

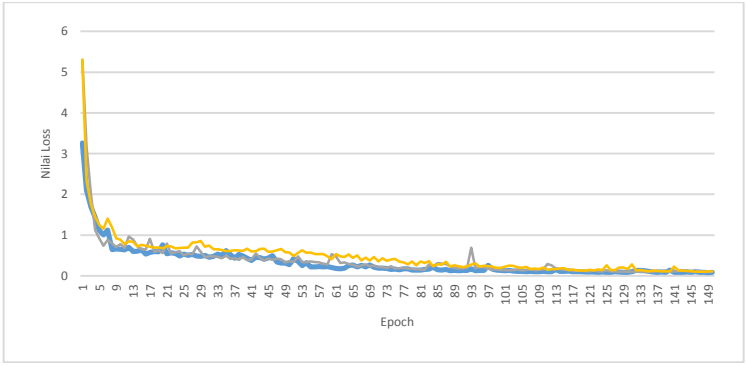
$$Y_{2,1} = \frac{1}{1+e^{-X_{2,1}}}$$

$$\vdots$$

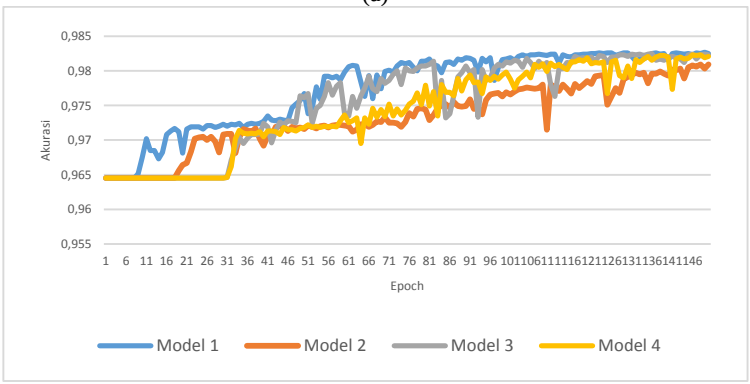
$$Y_{256,256} = \frac{1}{1+e^{-X_{256,256}}}$$

Diperoleh nilai untuk masing-masing *pixel* dengan interval nilai 0 hingga 1 sehingga diperlukan pembulatan nilai untuk memperoleh kelas untuk masing-masing *pixel.* Nilai 1 menunjukkan ROI pada citra MRI yang merupakan tumor otak, sedangkan nilai 0 menunjukkan *pixel* bukan merupakan ROI atau daerah non-tumor otak. Hasil pengklasifikasian untuk setiap *epoch* atau iterasi dilakukan penghitungan *loss* atau *error* serta akurasi dengan pembanding yang digunakan adalah *ground truth.* Rangkaian ini diulang sesuai dengan jumlah *epoch* yang telah ditetapkan yaitu sebesar 150 kali.

Pada model 2 hampir sama dengan model 1 tetapi jumlah *convolution block* sebanyak 4 pada tahap *encoder* sehingga dapat menurunkan ukuran gambar hingga 1/16 dari ukuran original. Pada model 3 dan 4 menggunakan dua *convolution layer* untuk setiap *convolution block*. Untuk seluruh model pada bagian *encoder* dan *decoder* menggunakan *padding* "*same*" untuk seluruh *convolution layer*. Sedangkan *activation function* yang digunakan adalah ReLU dan *sigmoid* serta optimasi adam untuk *update weight* dan *bias.* Pengoptimasian *weight* dan *bias* pada model merupakan optimasi parameter. Pengoptimasi parameter menggunakan metode adam ditetapkan nilai dari *learning rate* sebesar 0,0001, β_1=0,9 dan β_2= 0,999. Dilakukan operasi *backpropagation* untuk setiap kali *update* pada setiap *epoch* dari hasil penghitungan dari akurasi dan *loss* dari data *training* dan validasi. Sehingga nilai dari loss dan akurasi pada model untuk setiap *epoch* terjadi perbaikan. Gambar 4.10 adalah grafik nilai *loss* dan akurasi dari data validasi untuk setiap model.



(a)



(b)

Gambar 4.10 Grafik Hasil (a) *loss* dan (b) akurasi untuk Validasi

Dari Gambar 4.10 diketahui nilai *loss* dari masing-masing model turun hingga mendekati nilai 0. Semakin kecil dari nilai loss yang dihasilkan maka akan semakin bagus pula hasil klasifikasi yang diperoleh. Nilai akurasi pada model 3 dan 4 mulai berubah sejak memasuki *epoch* ke-30 namun model 2 mengalami perubahan rubahan sejak *epoch* ke 19. Model 2 merupakan model yang paling kecil perkembangannya jika dilihat dari nilai akurasi dari *training*. Model 1 memiliki kecenderungan nilai akurasi lebih baik jika dibandingkan model lain untuk setiap *epoch*. Tabel 4.4

merupakan hasil *loss* dan akurasi pada data *training* dan validasi pada *epoch* terakhir.

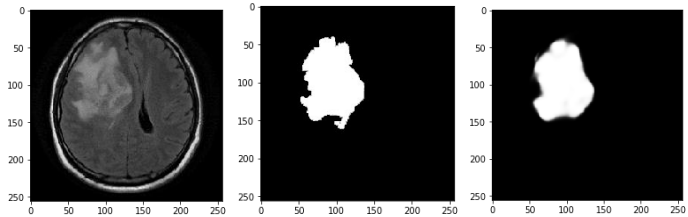
Tabel 4.4 Perbandingan *Loss* dan Akurasi pada setiap model

	Training		***Validation***	
	Loss	**Akurasi**	***Loss***	**Akurasi**
Model 1	**0,1800**	**0,9711**	**0,0877**	**0,9825**
Model 2	0,3055	0,969	0,1695	0,9809
Model 3	0,1935	0,9708	0,1035	0,9823
Model 4	0,2062	0,9706	0,1072	0,9821

Jika dilihat dari Tabel 4.4 model yang memiliki *loss* terkecil pada data *training* maupun data validasi adalah model 1 dengan *loss* sebesar 0,18 pada data *training* dan 0,0877 pada data validasi. Nilai akurasi terbesar juga dihasilkan oleh model 1 dengan akurasi pada data *training* sebesar 97,11% dan pada data validasi sebesar 98,25%. Model terbaik pada penelitian ini adalah pada model 1 dengan *loss* terkecil dan akurasi terbesar.

4.4 Segmentasi Citra MRI Tumor Otak

Dari hasil *running* data *training* dan data validasi diperoleh model 1 merupakan model terbaik. Model dan parameter yang didapat digunakan sebagai acuan untuk membuat segmentasi citra MRI tumor otak. Berikut merupakan hasil segmentasi pada data salah satu data *testing*.

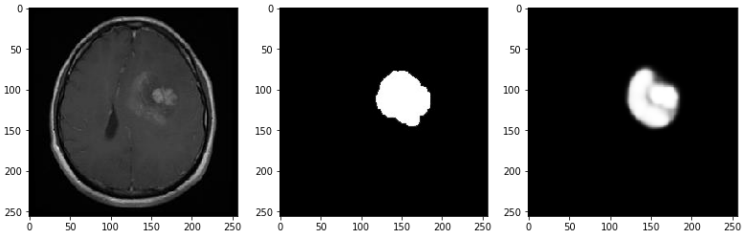


(a) (b) (c)

Gambar 4.11 citra MRI T2 Flair (a) *Ground truth* (b) dan hasil Segmentasi (c)

Pada citra MRI T2 *Flair* ROI tumor otak cenderung berwarna putih. Gambar 4.11 merupakan gambar citra MRI T2 *Flair* (a), g*round truth* ROI tumor otak dari ahli sebagai target (b) dan hasil segementasi setelah diolah menggunakan model 1. Terlihat

Fully convolutional network dapat menangkap ROI pada citra T2 *Flair* dengan baik. Pada citra MRI T1-C cenderung berwarna lebih gelap pada ROI tumor otak dibandingkan dengan MRI T2 *flair*. Gambar 4.12 menunjukkan citra MRI T1-C (a), g*round truth* ROI tumor otak dari ahli sebagai target (b) dan hasil segementasi setelah diolah menggunakan model 1. Contoh hasil segmentasi pada kasus T1-C dan T2 *flair* model 1 dapat menangkap ROI dari tumor otak. Model 1 dapat menangkap ROI tumor otak yang terletak pada bagian kiri maupun bagian kanan otak. Berikut merupakan hasil CCR pada ke-12 data *training* (Tabel 4.5).



(a) (b) (c)

Gambar 4.12 citra MRI T1-C (a) *Ground truth* (b) dan hasil Segmentasi (c)

Pada Tabel 4.5 hasil CCR dari seluruh data testing. T*esting* 1 hingga 6 merupakan tumor otak yang berada sisi kanan otak manusia, sedangkan *testing* 7 hingga 12 memiliki tumor otak disisi sebaliknya.

Tabel 4.5 Nilai CCR setiap data testing

Testing	**CCR**	***Testing***	**CCR**
1	0,963806	7	0,957581
2	0,975830	8	0,936691
3	0,931458	9	0,917374
4	0,957245	10	0,951462
5	0,900574	11	0,976624
6	0,906006	12	0,983017

Data *testing* ganjil merupakan citra MRI T2 *flair* sedangkan data *testing* genap merupakan citra MRI T1-C. Hasil segmentasi sudah mendekati *ground truth* dengan nilai rata-rata CCR pada

citra MRI T2 *flair* sebesar 0,9412 atau 94,12% sedangkan nilai rata-rata CCR pada citra MRI T1-C sebesar 0,9517 atau 95,17%. Setelah dilakukan pengujian CCR pada seluruh data *testing* diperoleh nilai rata-rata CCR adalah 94,65%.

(Halaman ini sengaja dikosongkan)

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan analisis dan pembahasan yang telah dilakukan pada bab 4, maka diperoleh kesimpulan sebagai berikut.

1. Dari empat model yang telah diuji diketahui bahwa model 1 yaitu model dengan ukuran citra diperkecil hingga 1/32 kali atau dengan digunakan 5 *convolution block* pada tahap *encoder* serta satu *convolution layer* untuk setiap *block.*
2. Hasil segmentasi pada data *testing* menghasilkan kecenderungan mendekati target ROI dari setiap citra MRI tumor otak. Hasil segmentasi dari 12 data *testing* diperoleh rata-rata nilai CCR sebesar 94,65%. Model dapat menangkap ROI tumor otak baik pada posisi kiri maupun pada sisi kanan otak.

5.2 Saran

Berdasarkan kesimpulan yang diperoleh, dapat dirumuskan saran sebagai pertimbangan penelitian selanjutnya adalah sebagai berikut,

1. Sebaiknya dalam penggunaan FCN digunakan juga R-CNN sebagai penetu daerah yang digunakan sebagai *input* agar data tidak besar dan semakin cepat dalam *running* program.
2. Dapat digunakan lebih banyak data sehingga "*machine*" dapat belajar pola-pola serta dapat diaplikasikan pada berbagai jenis kasus.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

Balafar, M.A., Ramli, A.R., Saripan, M.I., dan Mashohor, S. (2010). Review of brain MRI image segmentation methods. *Artif. Intell*. Rev. 33, 261–274. doi:10.1007/s10462-010-9155-0

Han, J., Kamber, M. dan Pei, J. (2013). *Data mining: Concepts and Techniques (Third Edition)*. SanFransisco: Morgan Kaufmann.

LeCun, Y., dan Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *Conference: The Handbook of Brain Theory and Neural Networks*. 255-258.

McAndrew, A. (2016), *A Computational Introduction to Digital Image Processing (Second Edition)*. Melbourne: CRC Press.

Naceur, M. B., Saouli, R., Akil, M., dan Kachouri, R. (2018), Fully Automatic Brain Tumor Segmentation using End-To-End Incremental Deep Neural Networks in MRI images. *Computer Methods and Programs in Biomedicine*. 166, 39–49.

Ostrom, Q. T., Gittleman, H., Truitt, G., Boscia, A., Kruchko, C., dan Barnholtz-Sloan, J. S. (2018). CBTRUS Statistical Report: Primary Brain and Other Central Nervous System Tumors Diagnosed in the United States in 2011–2015. *Neuro-Oncology*, 20(4), 1–86.

Pravitasari, A. A., Islamiyah, M. I., Iriawan, N., Irhamah, Fithriasari, K., Purnami, S. W., dan Ferriastuti, W., (2018). Fernandez-Steel Skew Normal (FSSN) Mixture Model Using Bayesian Approach for MRI-Based Brain Tumor Segmentation. Dipresentasikan pada: *International Seminar on Mathematics in Industry & International Conference on Theoretical and Applied Statistics (ISMI-ICTAS 2018)*, status diterbitkan: under review.

Pravitasari, A. A., Qonita, S. F., Iriawan, N., Irhamah, Fithriasari, K., Purnami, S. W., dan Ferriastuti, W., (2018). Segmentasi Citra Mri Tumor Otak Menggunakan Gaussian Mixture Model (GMM) dan Hybrid Gaussian Mixture Model –

Spatially Variant Finite Mixture Model (GMM-SVFMM) dengan Algoritma Expectation-Maximization (EM). Dipresentasikan pada: *International Seminar on Mathematics in Industry & International Conference on Theoretical and Applied Statistics (ISMI-ICTAS 2018),* status diterbitkan: under review.

Pravitasari, A. A., Safa, M. A. I., Iriawan, N., Irhamah, Fithriasari, K., Purnami, S. W., dan Ferriastuti, W., (2018). Segmentasi Citra MRI Tumor Otak Menggunakan Modified Stable Student-t Burr (MSTBurr) Mixture Model dengan Pendekatan Bayesian. Dipresentasikan pada: *International Seminar on Mathematics in Industry & International Conference on Theoretical and Applied Statistics (ISMI-ICTAS 2018),* status diterbitkan: under review..

Pravitasari, A. A., Solichah, S. A. N., Iriawan, N., Irhamah, Fithriasari, K., Purnami, S. W., dan Ferriastuti, W., (2018). Segmentasi Citra Otak Magnetic Resonance Imaging (MRI) Menggunakan Gaussian Mixture Model (GMM) dengan Pendekatan Expectation Maximization (EM) dalam Pembentukan Citra 3 Dimensi. Dipresentasikan pada: *International Seminar on Mathematics in Industry & International Conference on Theoretical and Applied Statistics (ISMI-ICTAS 2018),* status diterbitkan: under review.

Rawat, W., dan Wang, Z. (2017). Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Computation.* 29(9), 2352-2449.

RS Premier Surabya. (2017) *Rumah Sakit Premier Surabaya*. diaskses pada 15 Februari 2019 dari Deteksi Lebih Akurat dengan MRI 3 TESLA: http://rs-premiersurabaya.com/ deteksi-lebih-akurat-menggunakan-mri-3-tesla/.

Soesanti, I., Susanto, A., Widodo, T. S., dan Tjokronegoro, M. (2010). Segmentasi Citra Adaptif Berbasis Logika Fuzzy Teroptimasi untuk Analisis Citra Medis. *Forum Teknik*, 33(2), 89-96.

Stark. D. D, (1988), *Magnetic Resonance Imaging*, Toronto: The CV Mosby Company.

Sutoyo, T. D., Mulyanto, E., dan Suhartono, V. (2009). *Teori Pengolahan Citra Digital*. Yogyakarta: Andi.

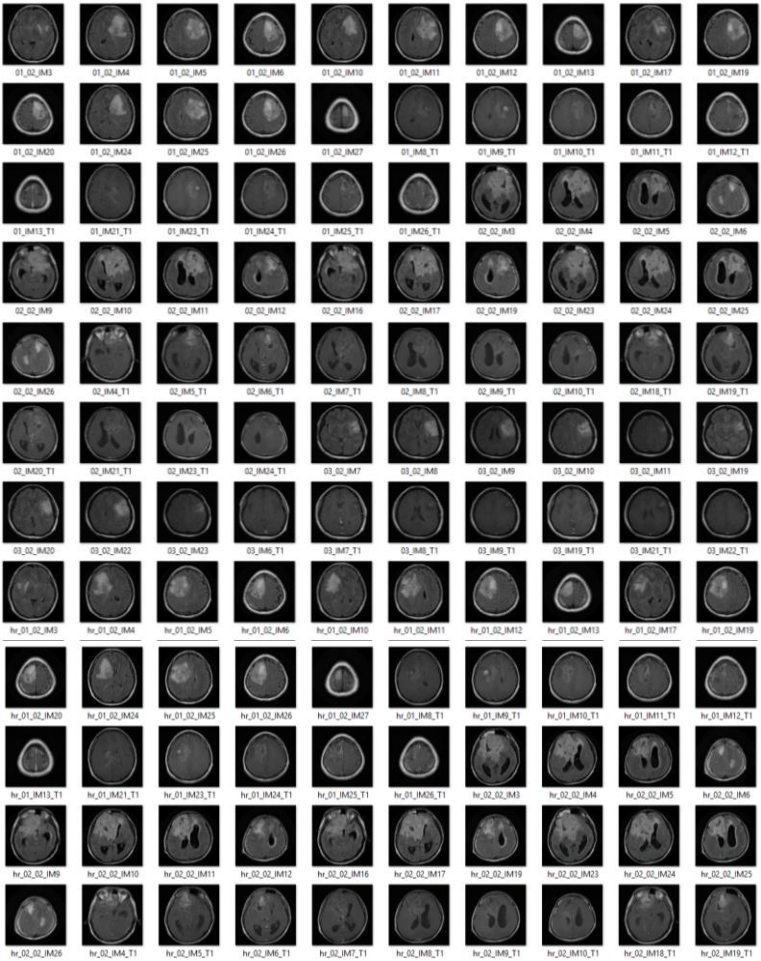
Yueniwati, Y. (2017). *Pencitraan pada Tumor Otak Modalitas dan Interpertasinya*, Malang: UB press.

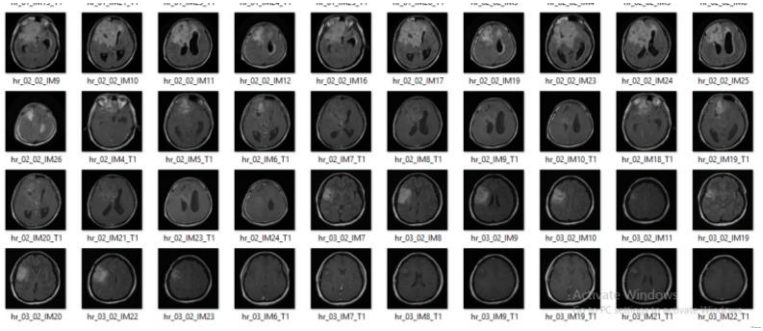
Zhang, A., Lipton,. Z. C., Li, M., dan Smola, A. J., (2019). *Dive into Deep Learning.* Diakses pada tanggal 27 Mei 2019 dari dive into deep learning: https://www.d2l.ai.

(Halaman ini sengaja dikosongkan)

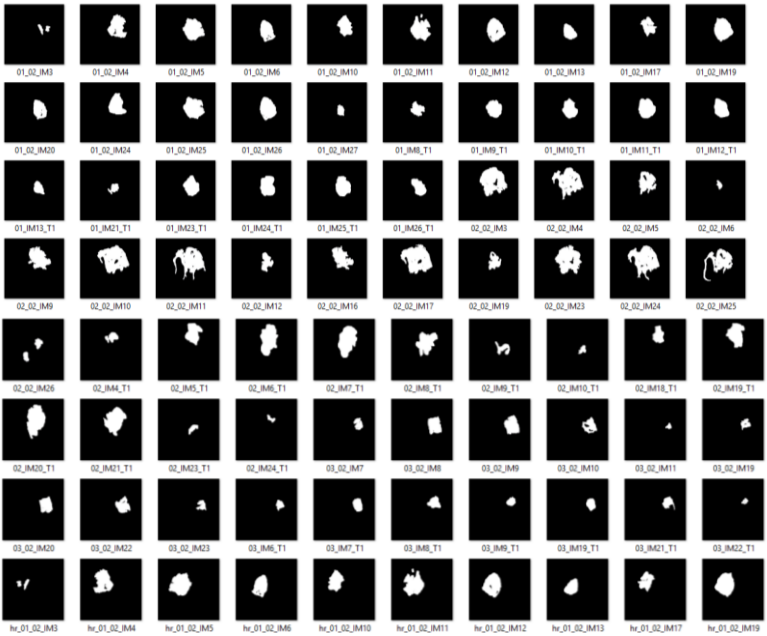
LAMPIRAN

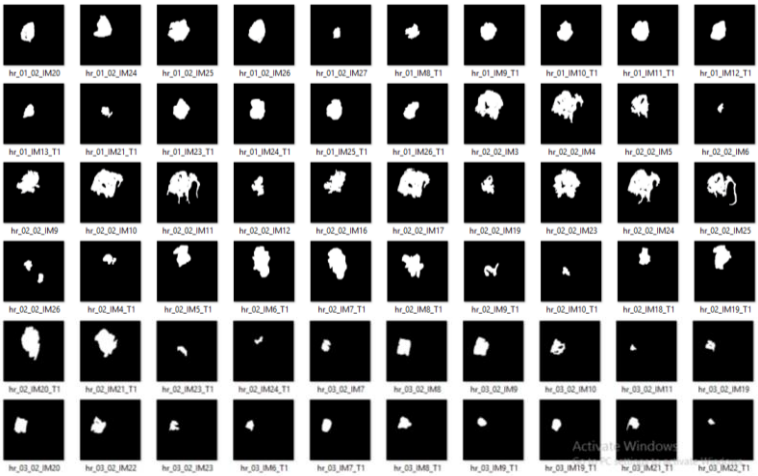
Lampiran 1. Data input citra MRI tumor otak





Lampiran 2. Data ROI *ground truth* citra MRI tumor otak





Lampiran 3. Tabel hasil akurasi dan *loss* dari setiap *epoch* pada model 1

Epoch	Loss	Acc	Val_loss	Val_acc	Epoch	Loss	Acc	Val_loss	Val_acc
1	8,23	0,70	3,26	0,96	76	0,32	0,97	0,15	0,98
2	4,61	0,93	2,10	0,96	77	0,31	0,97	0,17	0,98
3	3,85	0,93	1,69	0,96	78	0,31	0,97	0,17	0,98
4	3,07	0,93	1,44	0,96	79	0,31	0,97	0,14	0,98
5	2,25	0,93	1,11	0,96	80	0,30	0,97	0,14	0,98
6	1,76	0,93	1,01	0,96	81	0,31	0,97	0,14	0,98
7	1,61	0,93	1,12	0,96	82	0,29	0,97	0,16	0,98
8	1,49	0,93	0,64	0,96	83	0,32	0,97	0,17	0,98
9	1,42	0,93	0,66	0,97	84	0,33	0,97	0,21	0,98
10	1,39	0,94	0,65	0,97	85	0,30	0,97	0,15	0,98
11	1,43	0,94	0,63	0,97	86	0,30	0,97	0,15	0,98
12	1,45	0,94	0,70	0,97	87	0,29	0,97	0,16	0,98
13	1,35	0,94	0,60	0,97	88	0,30	0,97	0,13	0,98
14	1,26	0,95	0,61	0,97	89	0,29	0,97	0,13	0,98
15	1,21	0,95	0,63	0,97	90	0,30	0,97	0,12	0,98

Epoch	Loss	Acc	Val_loss	Val_acc	Epoch	Loss	Acc	Val_loss	Val_acc
16	1,17	0,95	0,53	0,97	91	0,29	0,97	0,13	0,98
17	1,12	0,95	0,57	0,97	92	0,29	0,97	0,13	0,98
18	1,12	0,95	0,60	0,97	93	0,32	0,97	0,17	0,98
19	1,12	0,95	0,59	0,97	94	0,30	0,97	0,13	0,98
20	1,23	0,94	0,76	0,97	95	0,29	0,97	0,14	0,98
21	1,16	0,95	0,54	0,97	96	0,29	0,97	0,13	0,98
22	1,00	0,95	0,57	0,97	97	0,32	0,97	0,26	0,98
23	0,98	0,95	0,55	0,97	98	0,34	0,97	0,16	0,98
24	0,99	0,95	0,48	0,97	99	0,38	0,97	0,14	0,98
25	1,00	0,95	0,54	0,97	100	0,32	0,97	0,13	0,98
26	0,94	0,95	0,50	0,97	101	0,29	0,97	0,13	0,98
27	0,89	0,96	0,53	0,97	102	0,26	0,97	0,13	0,98
28	0,86	0,96	0,49	0,97	103	0,25	0,97	0,12	0,98
29	0,90	0,96	0,48	0,97	104	0,24	0,97	0,11	0,98
30	0,83	0,96	0,50	0,97	105	0,23	0,97	0,11	0,98
31	0,82	0,96	0,45	0,97	106	0,23	0,97	0,11	0,98
32	0,81	0,96	0,48	0,97	107	0,22	0,97	0,11	0,98
33	0,77	0,96	0,53	0,97	108	0,22	0,97	0,10	0,98
34	0,78	0,96	0,50	0,97	109	0,22	0,97	0,10	0,98
35	0,79	0,96	0,62	0,97	110	0,22	0,97	0,11	0,98
36	0,77	0,96	0,53	0,97	111	0,23	0,97	0,10	0,98
37	0,77	0,96	0,43	0,97	112	0,24	0,97	0,10	0,98
38	0,72	0,96	0,52	0,97	113	0,28	0,97	0,15	0,98
39	0,73	0,96	0,49	0,97	114	0,28	0,97	0,11	0,98
40	0,67	0,96	0,44	0,97	115	0,25	0,97	0,11	0,98
41	0,66	0,96	0,38	0,97	116	0,26	0,97	0,12	0,98
42	0,73	0,96	0,45	0,97	117	0,23	0,97	0,10	0,98
43	0,67	0,96	0,45	0,97	118	0,22	0,97	0,10	0,98
44	0,68	0,96	0,41	0,97	119	0,21	0,97	0,10	0,98
45	0,66	0,96	0,44	0,97	120	0,22	0,97	0,10	0,98

Epoch	Loss	Acc	Val_loss	Val_acc	Epoch	Loss	Acc	Val_loss	Val_acc
46	0,61	0,96	0,49	0,97	121	0,22	0,97	0,09	0,98
47	0,65	0,96	0,34	0,97	122	0,21	0,97	0,10	0,98
48	0,59	0,96	0,32	0,98	123	0,21	0,97	0,09	0,98
49	0,59	0,96	0,31	0,98	124	0,20	0,97	0,09	0,98
50	0,57	0,96	0,28	0,98	125	0,20	0,97	0,09	0,98
51	0,53	0,96	0,43	0,97	126	0,20	0,97	0,09	0,98
52	0,53	0,96	0,35	0,97	127	0,20	0,97	0,10	0,98
53	0,50	0,96	0,26	0,98	128	0,22	0,97	0,09	0,98
54	0,48	0,97	0,32	0,98	129	0,21	0,97	0,09	0,98
55	0,47	0,97	0,22	0,98	130	0,19	0,97	0,09	0,98
56	0,44	0,97	0,22	0,98	131	0,20	0,97	0,10	0,98
57	0,48	0,97	0,23	0,98	132	0,22	0,97	0,13	0,98
58	0,43	0,97	0,22	0,98	133	0,22	0,97	0,13	0,98
59	0,41	0,97	0,23	0,98	134	0,26	0,97	0,13	0,98
60	0,40	0,97	0,20	0,98	135	0,23	0,97	0,11	0,98
61	0,39	0,97	0,18	0,98	136	0,21	0,97	0,09	0,98
62	0,38	0,97	0,17	0,98	137	0,21	0,97	0,09	0,98
63	0,38	0,97	0,19	0,98	138	0,20	0,97	0,09	0,98
64	0,40	0,97	0,25	0,98	139	0,21	0,97	0,09	0,98
65	0,40	0,97	0,26	0,98	140	0,21	0,97	0,14	0,98
66	0,43	0,97	0,22	0,98	141	0,21	0,97	0,09	0,98
67	0,45	0,97	0,26	0,98	142	0,21	0,97	0,08	0,98
68	0,61	0,96	0,22	0,98	143	0,21	0,97	0,09	0,98
69	0,68	0,96	0,27	0,98	144	0,21	0,97	0,09	0,98
70	0,48	0,96	0,21	0,98	145	0,20	0,97	0,09	0,98
71	0,41	0,97	0,19	0,98	146	0,21	0,97	0,10	0,98
72	0,38	0,97	0,19	0,98	147	0,20	0,97	0,09	0,98
73	0,35	0,97	0,18	0,98	148	0,19	0,97	0,09	0,98
74	0,35	0,97	0,16	0,98	149	0,19	0,97	0,08	0,98
75	0,34	0,97	0,16	0,98	150	0,18	0,97	0,09	0,98

Lampiran 4. *Syntax Jupyter Notebook import* pacage

```
import os
import sys
import random

import numpy as np
import cv2
import matplotlib.pyplot as plt

import tensorflow as tf
import pydicom
from tensorflow import keras
from medpy.io import load
from pydicom.data import get_testdata_files
from pydicom.filereader import read_dicomdir
from skimage.color import rgb2gray
from keras.callbacks import ReduceLROnPlateau
from keras.optimizers import Adam
```

Lampiran 5. *Syntax Jupyter Notebook seeding*

```
seed = 2019
random.seed = seed
np.random.seed = seed
tf.seed = seed
```

Lampiran 6. *Syntax Jupyter Notebook* fungsi Inisiasi data

```
class DataGen(keras.utils.Sequence):
    def __init__(self, ids, path, batch_size=3,
image_size=256):
        self.ids = ids
        self.path = path
        self.batch_size = batch_size
        self.image_size = image_size
        self.on_epoch_end()
```

Lampiran 7. *Syntax Jupyter Notebook load data*

```
def __load__(self, id_name):
        image_path = os.path.join(self.path,
id_name, "images", id_name) + ".dcm"
        mask_path = os.path.join(self.path,
id_name, "masks", id_name) +".jpg"

        ## Reading Image
        image = pydicom.dcmread(image_path)
        image = image.pixel_array
        mask = cv2.imread(mask_path, 0)
        mask.resize(256, 256, 1)

        image = image/255.0
        mask = mask/255.0

        return image, mask
```

Lampiran 8. *Syntax Jupyter Notebook* mengambil data

```
    def __getitem__(self, index):
        if(index+1)*self.batch_size >
len(self.ids):
            self.batch_size = len(self.ids) -
index*self.batch_size
```

```
        files_batch =
self.ids[index*self.batch_size :
(index+1)*self.batch_size]

        image = []
        mask  = []

        for id_name in files_batch:
            _img, _mask = self.__load__(id_name)
            image.append(_img)
            mask.append(_mask)

        image = np.array(image)
        mask  = np.array(mask)
        return image, mask

def on_epoch_end(self):
        pass

    def __len__(self):
        return
int(np.ceil(len(self.ids)/float(self.batch_size))
)
```

Lampiran 9. *Syntax Jupyter Notebook Running inisiasi*

```
image_size = 256
train_path = "F:/Semester 8/Tugas Akhir/TRY"
epochs = 5
batch_size = 3

## Training Ids
train_ids = next(os.walk(train_path))[1]

## Validation Data Size
val_data_size = 10

valid_ids = train_ids[:val_data_size]
train_ids = train_ids[val_data_size:]
```

Lampiran 10. *Syntax Jupyter Notebook* mengeluarkan ukuran matrix data

```
gen = DataGen(train_ids, train_path,
batch_size=batch_size, image_size=image_size)
x, y = gen.__getitem__(0)
print(x.shape, y.shape)
```

Lampiran 11. *Syntax Jupyter Notebook* membuat model UNet FCN

```
def down_block(x, filters, kernel_size=(3, 3),
padding="same", strides=1):
    c = keras.layers.Conv2D(filters, kernel_size,
padding=padding, strides=strides,
activation="relu")(x)
    c = keras.layers.Conv2D(filters, kernel_size,
padding=padding, strides=strides,
activation="relu")(c)
    p = keras.layers.MaxPool2D((2, 2), (2, 2))(c)
    return c, p

def up_block(x, skip, filters, kernel_size=(3,
3), padding="same", strides=1):
    us = keras.layers.UpSampling2D((2, 2))(x)
    concat = keras.layers.Concatenate()([us,
skip])
    c = keras.layers.Conv2D(filters, kernel_size,
padding=padding, strides=strides,
activation="relu")(concat)
    c = keras.layers.Conv2D(filters, kernel_size,
padding=padding, strides=strides,
activation="relu")(c)
    return c

def bottleneck(x, filters, kernel_size=(3, 3),
padding="same", strides=1):
    c = keras.layers.Conv2D(filters, kernel_size,
padding=padding, strides=strides,
activation="relu")(x)
    c = keras.layers.Conv2D(filters, kernel_size,
padding=padding, strides=strides,
activation="relu")(c)
    return c
```

```
def UNet():
    f = [16, 32, 64, 128, 256, 512]
    data_format='channels_last'
    inputs = keras.layers.Input((image_size,
image_size, 3))

    p0 = inputs
    c1, p1 = down_block(p0, f[0]) #128 -> 64
    c2, p2 = down_block(p1, f[1]) #64 -> 32
    c3, p3 = down_block(p2, f[2]) #32 -> 16
    c4, p4 = down_block(p3, f[3]) #16->8
    c5, p5 = down_block(p4, f[4])

    bn = bottleneck(p5, f[5])

    u1 = up_block(bn, c5, f[4]) #8 -> 16
    u2 = up_block(u1, c4, f[3]) #16 -> 32
    u3 = up_block(u2, c3, f[2]) #32 -> 64
    u4 = up_block(u3, c2, f[1]) #64 -> 128
    u5 = up_block(u4, c1, f[0])

    outputs = keras.layers.Conv2D(1, (1, 1),
padding="same", activation="sigmoid")(u5)
    model = keras.models.Model(inputs, outputs)
    return model

  model = UNet()
  adams = keras.optimizers.Adam(lr=0.001,
beta_1=0.9, beta_2=0.999, epsilon=None,
decay=0.0, amsgrad=False)
  model.compile(optimizer=adams ,
loss="binary_crossentropy", metrics=["acc"])
  model.summary()
```

Lampiran 12. *Syntax Jupyter Notebook* running model

```
  train_gen = DataGen(train_ids, train_path,
image_size=image_size, batch_size=batch_size)
  valid_gen = DataGen(valid_ids, train_path,
image_size=image_size, batch_size=batch_size)

  train_steps = len(train_ids)//batch_size
  valid_steps = len(valid_ids)//batch_size

  model.fit_generator(train_gen,
validation_data=valid_gen,
steps_per_epoch=train_steps,
validation_steps=valid_steps,
                      epochs=epochs)
```

Lampiran 13. *Syntax Jupyter Notebook* simpan bobot model

```
## Save the Weights
model.save_weights("UNetW_epoch5_run22_batch3_1.h
5")
## Dataset for prediction
x, y = valid_gen.__getitem__(1)
result = model.predict(x)
```

Lampiran 14. *Syntax Jupyter Notebook* mengeluarkan gambar *ground truth* dan hasil segmentasi dari data validasi

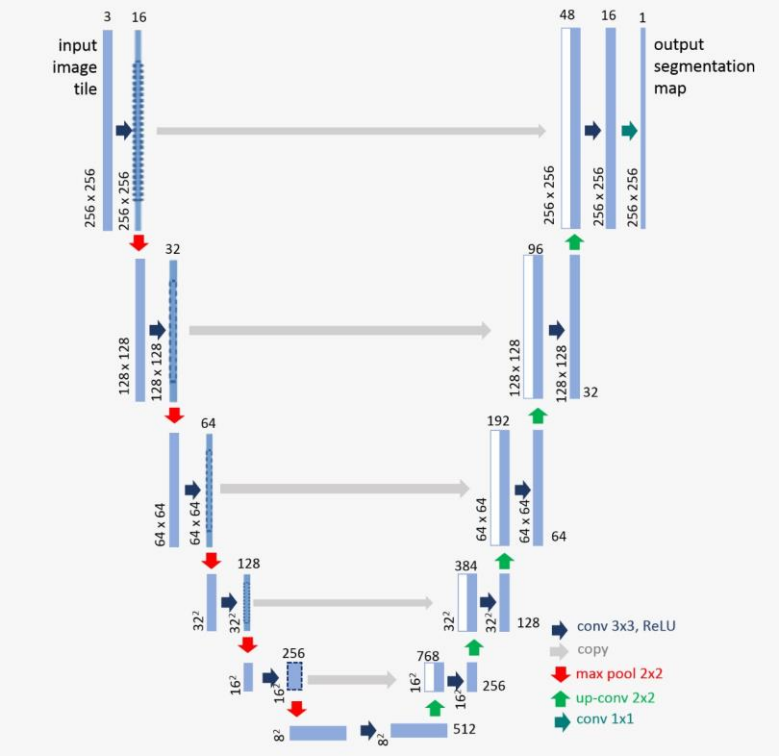
```
fig = plt.figure()
fig.subplots_adjust(hspace=0.4, wspace=0.4)

ax = fig.add_subplot(1, 2, 1)
ax.imshow(np.reshape(y[0], (image_size,
image_size)), cmap="gray")

ax = fig.add_subplot(1, 2, 2)
ax.imshow(np.reshape(result[0], (image_size,
image_size)), cmap="gray")
```

Lampiran 15. Matriks gambar

$$\textit{Ground truth} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}$$

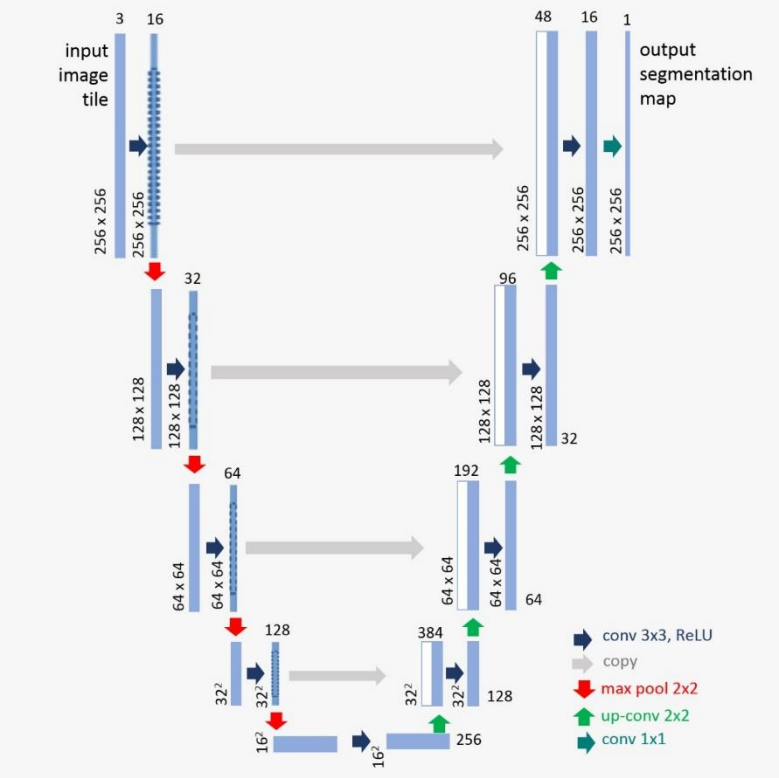
$$\text{MRI} = \begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}, \\ \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix} \end{bmatrix}$$

Lampiran 16. Struktur Model 1



```
Layer (type)                 Output Shape              Param #   Connected to
==================================================================================================
input_11 (InputLayer)        (None, 256, 256, 3)       0
--------------------------------------------------------------------------------------------------
conv2d_145 (Conv2D)          (None, 256, 256, 16)      448       input_11[0][0]
--------------------------------------------------------------------------------------------------
max_pooling2d_47 (MaxPooling2D) (None, 128, 128, 16)   0         conv2d_145[0][0]
--------------------------------------------------------------------------------------------------
conv2d_146 (Conv2D)          (None, 128, 128, 32)      4640      max_pooling2d_47[0][0]
--------------------------------------------------------------------------------------------------
max_pooling2d_48 (MaxPooling2D) (None, 64, 64, 32)     0         conv2d_146[0][0]
--------------------------------------------------------------------------------------------------
conv2d_147 (Conv2D)          (None, 64, 64, 64)        18496     max_pooling2d_48[0][0]
--------------------------------------------------------------------------------------------------
max_pooling2d_49 (MaxPooling2D) (None, 32, 32, 64)     0         conv2d_147[0][0]
--------------------------------------------------------------------------------------------------
conv2d_148 (Conv2D)          (None, 32, 32, 128)       73856     max_pooling2d_49[0][0]
--------------------------------------------------------------------------------------------------
max_pooling2d_50 (MaxPooling2D) (None, 16, 16, 128)    0         conv2d_148[0][0]
--------------------------------------------------------------------------------------------------
conv2d_149 (Conv2D)          (None, 16, 16, 256)       295168    max_pooling2d_50[0][0]
--------------------------------------------------------------------------------------------------
max_pooling2d_51 (MaxPooling2D) (None, 8, 8, 256)      0         conv2d_149[0][0]
--------------------------------------------------------------------------------------------------
conv2d_150 (Conv2D)          (None, 8, 8, 512)         1180160   max_pooling2d_51[0][0]
--------------------------------------------------------------------------------------------------
up_sampling2d_47 (UpSampling2D) (None, 16, 16, 512)    0         conv2d_150[0][0]
--------------------------------------------------------------------------------------------------
concatenate_47 (Concatenate) (None, 16, 16, 768)       0         up_sampling2d_47[0][0]
                                                                 conv2d_149[0][0]
--------------------------------------------------------------------------------------------------
conv2d_151 (Conv2D)          (None, 16, 16, 256)       1769728   concatenate_47[0][0]
--------------------------------------------------------------------------------------------------
up_sampling2d_48 (UpSampling2D) (None, 32, 32, 256)    0         conv2d_151[0][0]
--------------------------------------------------------------------------------------------------
concatenate_48 (Concatenate) (None, 32, 32, 384)       0         up_sampling2d_48[0][0]
                                                                 conv2d_148[0][0]
--------------------------------------------------------------------------------------------------
conv2d_152 (Conv2D)          (None, 32, 32, 128)       442496    concatenate_48[0][0]
--------------------------------------------------------------------------------------------------
up_sampling2d_49 (UpSampling2D) (None, 64, 64, 128)    0         conv2d_152[0][0]
--------------------------------------------------------------------------------------------------
concatenate_49 (Concatenate) (None, 64, 64, 192)       0         up_sampling2d_49[0][0]
                                                                 conv2d_147[0][0]
--------------------------------------------------------------------------------------------------
conv2d_153 (Conv2D)          (None, 64, 64, 64)        110656    concatenate_49[0][0]
--------------------------------------------------------------------------------------------------
up_sampling2d_50 (UpSampling2D) (None, 128, 128, 64)   0         conv2d_153[0][0]
--------------------------------------------------------------------------------------------------
oncatenate_50 (Concatenate)  (None, 128, 128, 96)      0         up_sampling2d_50[0][0]
                                                                 conv2d_146[0][0]
--------------------------------------------------------------------------------------------------
onv2d_154 (Conv2D)           (None, 128, 128, 32)      27680     concatenate_50[0][0]
--------------------------------------------------------------------------------------------------
_sampling2d_51 (UpSampling2D) (None, 256, 256, 32)     0         conv2d_154[0][0]
--------------------------------------------------------------------------------------------------
oncatenate_51 (Concatenate)  (None, 256, 256, 48)      0         up_sampling2d_51[0][0]
                                                                 conv2d_145[0][0]
--------------------------------------------------------------------------------------------------
onv2d_155 (Conv2D)           (None, 256, 256, 16)      6928      concatenate_51[0][0]
--------------------------------------------------------------------------------------------------
onv2d_156 (Conv2D)           (None, 256, 256, 1)       17        conv2d_155[0][0]
==================================================================================================
otal params: 3,930,273
ainable params: 3,930,273
on-trainable params: 0
--------------------------------------------------------------------------------------------------
```

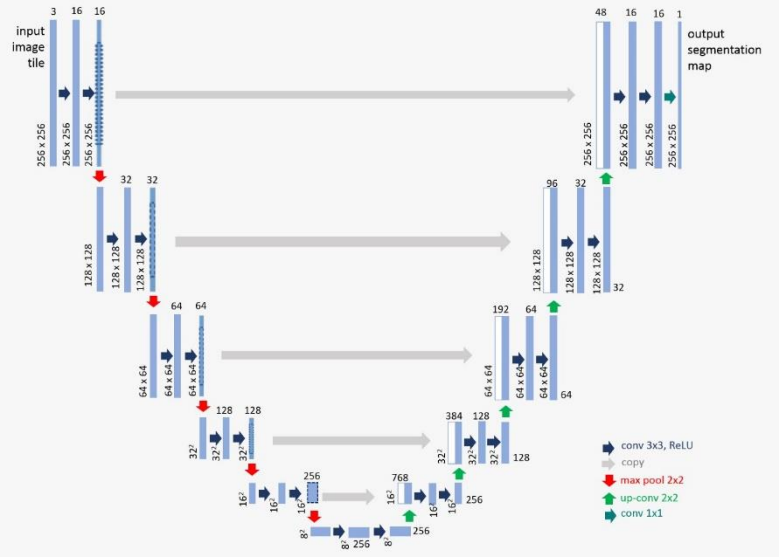
Lampiran 17. Struktur Model 2



Layer (type)	Output Shape	Param #	Connected to
input_10 (InputLayer)	(None, 256, 256, 3)	0	
conv2d_135 (Conv2D)	(None, 256, 256, 16)	448	input_10[0][0]
max_pooling2d_43 (MaxPooling2D)	(None, 128, 128, 16)	0	conv2d_135[0][0]
conv2d_136 (Conv2D)	(None, 128, 128, 32)	4640	max_pooling2d_43[0][0]
max_pooling2d_44 (MaxPooling2D)	(None, 64, 64, 32)	0	conv2d_136[0][0]
conv2d_137 (Conv2D)	(None, 64, 64, 64)	18496	max_pooling2d_44[0][0]
max_pooling2d_45 (MaxPooling2D)	(None, 32, 32, 64)	0	conv2d_137[0][0]
conv2d_138 (Conv2D)	(None, 32, 32, 128)	73856	max_pooling2d_45[0][0]
max_pooling2d_46 (MaxPooling2D)	(None, 16, 16, 128)	0	conv2d_138[0][0]
conv2d_139 (Conv2D)	(None, 16, 16, 256)	295168	max_pooling2d_46[0][0]
up_sampling2d_43 (UpSampling2D)	(None, 32, 32, 256)	0	conv2d_139[0][0]
concatenate_43 (Concatenate)	(None, 32, 32, 384)	0	up_sampling2d_43[0][0] conv2d_138[0][0]
conv2d_140 (Conv2D)	(None, 32, 32, 128)	442496	concatenate_43[0][0]
up_sampling2d_44 (UpSampling2D)	(None, 64, 64, 128)	0	conv2d_140[0][0]
concatenate_44 (Concatenate)	(None, 64, 64, 192)	0	up_sampling2d_44[0][0] conv2d_137[0][0]
conv2d_141 (Conv2D)	(None, 64, 64, 64)	110656	concatenate_44[0][0]
up_sampling2d_45 (UpSampling2D)	(None, 128, 128, 64)	0	conv2d_141[0][0]
concatenate_45 (Concatenate)	(None, 128, 128, 96)	0	up_sampling2d_45[0][0] conv2d_136[0][0]
conv2d_142 (Conv2D)	(None, 128, 128, 32)	27680	concatenate_45[0][0]
up_sampling2d_46 (UpSampling2D)	(None, 256, 256, 32)	0	conv2d_142[0][0]
:oncatenate_46 (Concatenate)	(None, 256, 256, 48)	0	up_sampling2d_46[0][0] conv2d_135[0][0]
:onv2d_143 (Conv2D)	(None, 256, 256, 16)	6928	concatenate_46[0][0]
:onv2d_144 (Conv2D)	(None, 256, 256, 1)	17	conv2d_143[0][0]

Total params: 980,385
Trainable params: 980,385
Non-trainable params: 0

Lampiran 18. Struktur Model 3

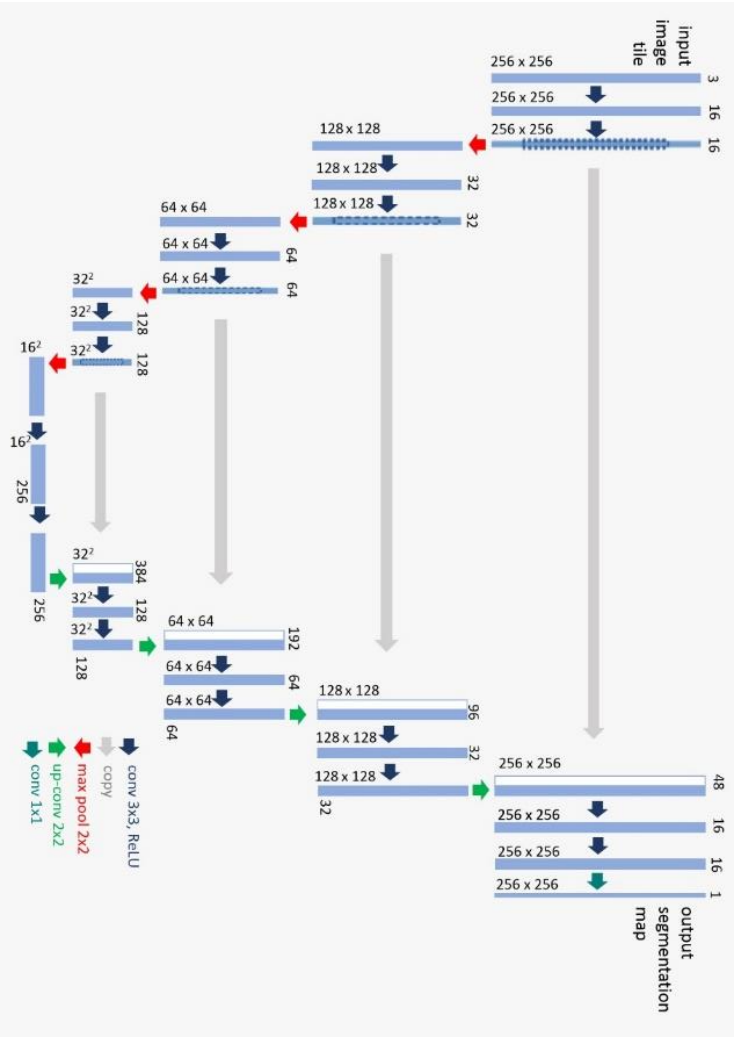


Layer (type)	Output Shape	Param #	Connected to
input_12 (InputLayer)	(None, 256, 256, 3)	0	
conv2d_157 (Conv2D)	(None, 256, 256, 16)	448	input_12[0][0]
conv2d_158 (Conv2D)	(None, 256, 256, 16)	2320	conv2d_157[0][0]
max_pooling2d_52 (MaxPooling2D)	(None, 128, 128, 16)	0	conv2d_158[0][0]
conv2d_159 (Conv2D)	(None, 128, 128, 32)	4640	max_pooling2d_52[0][0]
conv2d_160 (Conv2D)	(None, 128, 128, 32)	9248	conv2d_159[0][0]
max_pooling2d_53 (MaxPooling2D)	(None, 64, 64, 32)	0	conv2d_160[0][0]
conv2d_161 (Conv2D)	(None, 64, 64, 64)	18496	max_pooling2d_53[0][0]
conv2d_162 (Conv2D)	(None, 64, 64, 64)	36928	conv2d_161[0][0]
max_pooling2d_54 (MaxPooling2D)	(None, 32, 32, 64)	0	conv2d_162[0][0]
conv2d_163 (Conv2D)	(None, 32, 32, 128)	73856	max_pooling2d_54[0][0]
conv2d_164 (Conv2D)	(None, 32, 32, 128)	147584	conv2d_163[0][0]
max_pooling2d_55 (MaxPooling2D)	(None, 16, 16, 128)	0	conv2d_164[0][0]
conv2d_165 (Conv2D)	(None, 16, 16, 256)	295168	max_pooling2d_55[0][0]
conv2d_166 (Conv2D)	(None, 16, 16, 256)	590080	conv2d_165[0][0]
max_pooling2d_56 (MaxPooling2D)	(None, 8, 8, 256)	0	conv2d_166[0][0]

conv2d_167 (Conv2D)	(None, 8, 8, 512)	1180160	max_pooling2d_56[0][0]
conv2d_168 (Conv2D)	(None, 8, 8, 512)	2359808	conv2d_167[0][0]
up_sampling2d_52 (UpSampling2D)	(None, 16, 16, 512)	0	conv2d_168[0][0]
concatenate_52 (Concatenate)	(None, 16, 16, 768)	0	up_sampling2d_52[0][0] conv2d_166[0][0]
conv2d_169 (Conv2D)	(None, 16, 16, 256)	1769728	concatenate_52[0][0]
conv2d_170 (Conv2D)	(None, 16, 16, 256)	590080	conv2d_169[0][0]
up_sampling2d_53 (UpSampling2D)	(None, 32, 32, 256)	0	conv2d_170[0][0]
concatenate_53 (Concatenate)	(None, 32, 32, 384)	0	up_sampling2d_53[0][0] conv2d_164[0][0]
conv2d_171 (Conv2D)	(None, 32, 32, 128)	442496	concatenate_53[0][0]
conv2d_172 (Conv2D)	(None, 32, 32, 128)	147584	conv2d_171[0][0]
up_sampling2d_54 (UpSampling2D)	(None, 64, 64, 128)	0	conv2d_172[0][0]
concatenate_54 (Concatenate)	(None, 64, 64, 192)	0	up_sampling2d_54[0][0] conv2d_162[0][0]
conv2d_173 (Conv2D)	(None, 64, 64, 64)	110656	concatenate_54[0][0]
conv2d_174 (Conv2D)	(None, 64, 64, 64)	36928	conv2d_173[0][0]
up_sampling2d_55 (UpSampling2D)	(None, 128, 128, 64)	0	conv2d_174[0][0]
concatenate_55 (Concatenate)	(None, 128, 128, 96)	0	up_sampling2d_55[0][0] conv2d_160[0][0]
conv2d_175 (Conv2D)	(None, 128, 128, 32)	27680	concatenate_55[0][0]
conv2d_176 (Conv2D)	(None, 128, 128, 32)	9248	conv2d_175[0][0]
up_sampling2d_56 (UpSampling2D)	(None, 256, 256, 32)	0	conv2d_176[0][0]
concatenate_56 (Concatenate)	(None, 256, 256, 48)	0	up_sampling2d_56[0][0] conv2d_158[0][0]
conv2d_177 (Conv2D)	(None, 256, 256, 16)	6928	concatenate_56[0][0]
conv2d_178 (Conv2D)	(None, 256, 256, 16)	2320	conv2d_177[0][0]
conv2d_179 (Conv2D)	(None, 256, 256, 1)	17	conv2d_178[0][0]

Total params: 7,862,401
Trainable params: 7,862,401
Non-trainable params: 0

Lampiran 19. Struktur Model 4



Layer (type)	Output Shape	Param #	Connected to
input_13 (InputLayer)	(None, 256, 256, 3)	0	
conv2d_180 (Conv2D)	(None, 256, 256, 16)	448	input_13[0][0]
conv2d_181 (Conv2D)	(None, 256, 256, 16)	2320	conv2d_180[0][0]
max_pooling2d_57 (MaxPooling2D)	(None, 128, 128, 16)	0	conv2d_181[0][0]
conv2d_182 (Conv2D)	(None, 128, 128, 32)	4640	max_pooling2d_57[0][0]
conv2d_183 (Conv2D)	(None, 128, 128, 32)	9248	conv2d_182[0][0]
max_pooling2d_58 (MaxPooling2D)	(None, 64, 64, 32)	0	conv2d_183[0][0]
conv2d_184 (Conv2D)	(None, 64, 64, 64)	18496	max_pooling2d_58[0][0]
conv2d_185 (Conv2D)	(None, 64, 64, 64)	36928	conv2d_184[0][0]
max_pooling2d_59 (MaxPooling2D)	(None, 32, 32, 64)	0	conv2d_185[0][0]
conv2d_186 (Conv2D)	(None, 32, 32, 128)	73856	max_pooling2d_59[0][0]
conv2d_187 (Conv2D)	(None, 32, 32, 128)	147584	conv2d_186[0][0]
max_pooling2d_60 (MaxPooling2D)	(None, 16, 16, 128)	0	conv2d_187[0][0]
conv2d_188 (Conv2D)	(None, 16, 16, 256)	295168	max_pooling2d_60[0][0]
conv2d_189 (Conv2D)	(None, 16, 16, 256)	590080	conv2d_188[0][0]
up_sampling2d_57 (UpSampling2D)	(None, 32, 32, 256)	0	conv2d_189[0][0]
concatenate_57 (Concatenate)	(None, 32, 32, 384)	0	up_sampling2d_57[0][0] conv2d_187[0][0]
conv2d_190 (Conv2D)	(None, 32, 32, 128)	442496	concatenate_57[0][0]
conv2d_191 (Conv2D)	(None, 32, 32, 128)	147584	conv2d_190[0][0]
up_sampling2d_58 (UpSampling2D)	(None, 64, 64, 128)	0	conv2d_191[0][0]
concatenate_58 (Concatenate)	(None, 64, 64, 192)	0	up_sampling2d_58[0][0] conv2d_185[0][0]
conv2d_192 (Conv2D)	(None, 64, 64, 64)	110656	concatenate_58[0][0]

:onv2d_187 (Conv2D)	(None, 32, 32, 128)	147584	conv2d_186[0][0]
ıax_pooling2d_60 (MaxPooling2D)	(None, 16, 16, 128)	0	conv2d_187[0][0]
:onv2d_188 (Conv2D)	(None, 16, 16, 256)	295168	max_pooling2d_60[0][0]
:onv2d_189 (Conv2D)	(None, 16, 16, 256)	590080	conv2d_188[0][0]
ıp_sampling2d_57 (UpSampling2D)	(None, 32, 32, 256)	0	conv2d_189[0][0]
:oncatenate_57 (Concatenate)	(None, 32, 32, 384)	0	up_sampling2d_57[0][0] conv2d_187[0][0]
:onv2d_190 (Conv2D)	(None, 32, 32, 128)	442496	concatenate_57[0][0]
:onv2d_191 (Conv2D)	(None, 32, 32, 128)	147584	conv2d_190[0][0]
ıp_sampling2d_58 (UpSampling2D)	(None, 64, 64, 128)	0	conv2d_191[0][0]
:oncatenate_58 (Concatenate)	(None, 64, 64, 192)	0	up_sampling2d_58[0][0] conv2d_185[0][0]
:onv2d_192 (Conv2D)	(None, 64, 64, 64)	110656	concatenate_58[0][0]
:onv2d_193 (Conv2D)	(None, 64, 64, 64)	36928	conv2d_192[0][0]
ıp_sampling2d_59 (UpSampling2D)	(None, 128, 128, 64)	0	conv2d_193[0][0]
:oncatenate_59 (Concatenate)	(None, 128, 128, 96)	0	up_sampling2d_59[0][0] conv2d_183[0][0]
:onv2d_194 (Conv2D)	(None, 128, 128, 32)	27680	concatenate_59[0][0]
:onv2d_195 (Conv2D)	(None, 128, 128, 32)	9248	conv2d_194[0][0]
ıp_sampling2d_60 (UpSampling2D)	(None, 256, 256, 32)	0	conv2d_195[0][0]
:oncatenate_60 (Concatenate)	(None, 256, 256, 48)	0	up_sampling2d_60[0][0] conv2d_181[0][0]
:onv2d_196 (Conv2D)	(None, 256, 256, 16)	6928	concatenate_60[0][0]
:onv2d_197 (Conv2D)	(None, 256, 256, 16)	2320	conv2d_196[0][0]
:onv2d_198 (Conv2D)	(None, 256, 256, 1)	17	conv2d_197[0][0]

:otal params: 1,962,625
Trainable params: 1,962,625
Non-trainable params: 0

Lampiran 20. Pembobot untuk Model Terbaik

Pembobot untuk *Convolutional Layer* ke-1

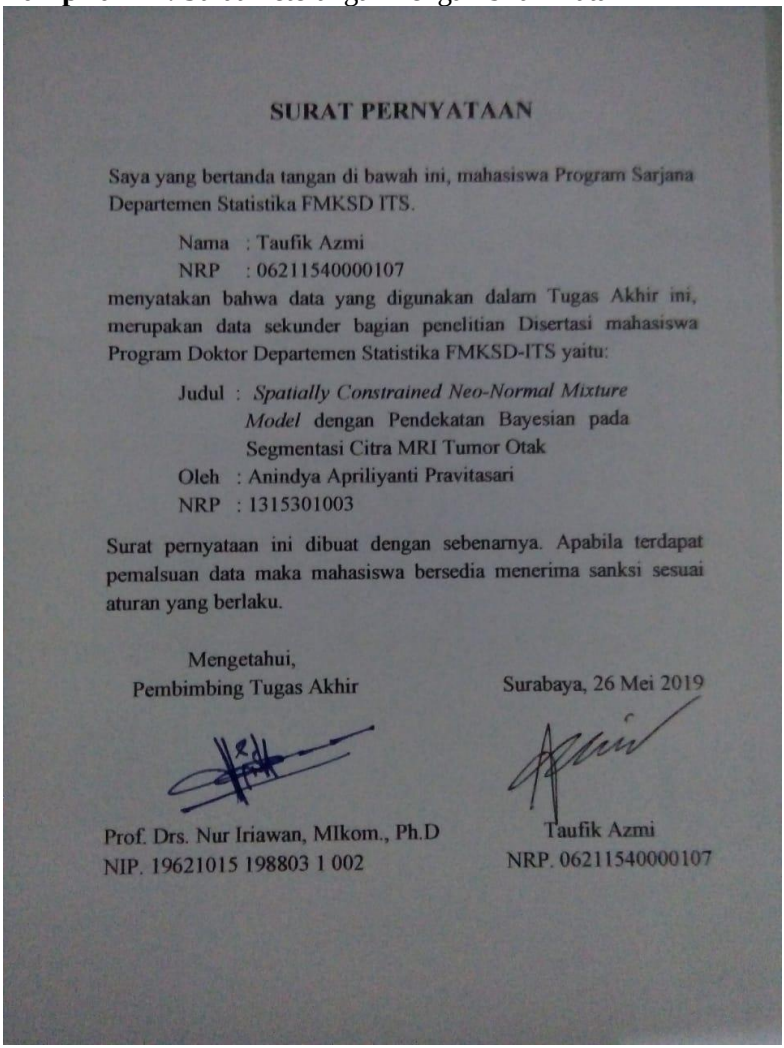
	1	2	3	…	16
bias	-0,00548	0,00861	-0,00297	…	0,00466

Filter	1	2	3
1	$\begin{bmatrix} -0,01304 & 0,08496 & 0,05443 \\ 0,16343 & 0,16381 & 0,15175 \\ 0,07392 & -0,04229 & 0,06072 \end{bmatrix}$	$\begin{bmatrix} 0,15916 & -0,11795 & 6,32101 \\ 0,04702 & 0,11148 & 0,09227 \\ 0,00659 & 0,06171 & 0,13466 \end{bmatrix}$	$\begin{bmatrix} -0,06126 & -0,06167 & 0,05309 \\ 0,04395 & -0,15623 & 0,09789 \\ -0,03681 & 0,08185 & -0,16235 \end{bmatrix}$
…	…	…	…
16	$\begin{bmatrix} 0,03779 & 0,12983 & -0,01291 \\ 0,00681 & -0,06969 & 0,02662 \\ 0,03947 & 0,05122 & -0,09632 \end{bmatrix}$	$\begin{bmatrix} -0,01355 & 0,12281 & -0,13653 \\ -0,08498 & -0,04290 & -0,09309 \\ 0,12093 & 0,04881 & -0,13270 \end{bmatrix}$	$\begin{bmatrix} -0,11217 & 0,05850 & 0,09234 \\ -0,03732 & -0,18307 & 0,18367 \\ -0,01172 & -0,01760 & -0,18755 \end{bmatrix}$

Pembobot untuk *Convolutional Layer* ke-12

Bias	Filter	
-0,00121	1	$[0,60877]$
	2	$[0,37943]$
	3	$[0,29601]$
	…	…
	16	$[-0,5331]$

Lampiran 21. Surat Keterangan Pengambilan Data



SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, mahasiswa Program Sarjana Departemen Statistika FMKSD ITS.

Nama : Taufik Azmi
NRP : 06211540000107

menyatakan bahwa data yang digunakan dalam Tugas Akhir ini, merupakan data sekunder bagian penelitian Disertasi mahasiswa Program Doktor Departemen Statistika FMKSD-ITS yaitu:

Judul : *Spatially Constrained Neo-Normal Mixture Model* dengan Pendekatan Bayesian pada Segmentasi Citra MRI Tumor Otak
Oleh : Anindya Apriliyanti Pravitasari
NRP : 1315301003

Surat pernyataan ini dibuat dengan sebenarnya. Apabila terdapat pemalsuan data maka mahasiswa bersedia menerima sanksi sesuai aturan yang berlaku.

Mengetahui,
Pembimbing Tugas Akhir

Prof. Drs. Nur Iriawan, MIkom., Ph.D
NIP. 19621015 198803 1 002

Surabaya, 26 Mei 2019

Taufik Azmi
NRP. 06211540000107

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Penulis bernama Taufik Azmi dilahirkan di Kota Bondowoso pada tanggal 25 Desember 1996 sebagai anak kedua dari dua bersaudara. Penulis merupakan putra kedua dari dua bersaudara. Ayah Saifudin dan Ibu Titiek Tjaturweni. Penulis menempuh pendidikan formal di SDN Kepuharjo 2 Lumajang, SMP 1 Sukodono Lumajang, dan SMA Negeri 2 Lumajang. Kemudian penulis diterima sebagai Mahasiswa jurusan Statistika FMIPA ITS melalui jalur SBMPTN pada tahun 2015 dengan NRP 1315100107. Selama masa perkuliahan, penulis memiliki beberapa pengalaman dalam berpartisipasi berbagai kepanitiaan diantaranya sebagai Anggota Divisi Perlengkapan dalam event tahunan PRS (Pekan Raya Statistika) 2017 dan 2018, yang merupakan kompetisi Nasional. Serta Anggota Divisi Perlengkapan dalam event ISCO. Mengadakan Tryout SMA melalui FORDA Lumajang yang ada di Surabaya. Penulis juga pernah mengikuti pelatihan yaitu LKMM-Pra-TD. Apabila pembaca ingin memberi kritik dan saran serta diskusi lebih lanjut mengenai Tugas Akhir ini, dapat menghubungi penulis melalui email taufikazmi251296@gmail.com.