Pemanfaatan 3D U-Net untuk Segmentasi 3 Dimensi Gelembung Penyebab Kanker Paru-paru (Nodule) pada Lapisan Citra CT Scan

Agung Pribadi, *Departemen Informatika, Institut Sains dan Teknologi Terpadu Surabaya*,
Anang Kukuh Adisusilo, *Departemen Informatika, Universitas Wijaya Kusuma Surabaya*.

***Abstrak*- Kanker paru-paru adalah salah satu tipe kanker yang mematikan di seluruh dunia. Proses pendeteksian manual memakan banyak waktu dan energi untuk dituangkan. Kerja keras sekalipun tidaklah cukup, maka telat dalam pendeteksian telah menjadi faktor utama dalam tingkat kesembuhan. Menggunakan[CT Scan adalah salah satu alat skrining yang potensial, tetapi program skrining otomatis dibutuhkan untuk menghemat waktu. Pendeteksian, menyegmentasi dan penggolongan nodule adalah tiga poin utama untuk membuat ini menjadi nyata. Oleh karena itu, dalam mengambil bagian untuk membuat program itu, tugas akhir ini akan melakukan fokus menyegmentasi nodule. Pada awal program, pertama dibutuhkan hasil CT Scan dan segmentasi nodulenya. Selain itu, juga dibutuhkan titik koordinat centroid dari nodule dari hasil scan tersebut. Setelah mengetahui titik centroid dari nodule, maka proses preprocessing akan menjadi mudah. Program akan mengambil daerah sekitar nodule dan nodule sebagai titik tengahnya, hasil dari proses ini adalah sebuah kubus yang memiliki ukuran yang sama untuk semua hasil preprocess yang sebelumnya sudah ditentukan. Setelah mendapatkan kubus daerah sekitar nodule, proses normalisasi akan dilakukan sehingga data siap untuk ditraning oleh neural network. 2 Model dibuat dengan arsitektur yang berbeda, seperti menggunakan U-Net biasa dan U-Net++. Hasil dari program akan menghasilkan skor Jaccard index dengan tingkat keakuratan tertinggi sebesar 79.8%. Didapatkan kesimpulan bahwa penggunaan U-Net biasa dan U-Net++ tidak memiliki perbedaan akurasi yang signifikan, meskipun memiliki komponen layer dan aktivasi yang sama.**

***Kata Kunci*- CT Scan, Kanker Paru-paru, Nodule, Segmentasi, U-Net.**

I. PENDAHULUAN

Dalam era digital sekarang ini, teknologi sudah melingkupi banyak bidang pekerjaan, salah satunya adalah bidang kesehatan. Dalam bidang kesehatan, peralatan kedokteran terus diperbarui untuk menambah fungsi alat ataupun meningkatkan performa alat. Dalam kegiatan operasi, dokter spesialis selalu melihat bagian mana yang perlu dibedah, dan melihat apakah kegiatan pembedahan perlu dilakukan atau tidak melalui hasil scanner. Dengan begitu dokter dapat bekerja secara efektif dan tepat sasaran.

Kanker adalah penyakit yang perlu ditangani dengan cepat untuk meningkatkan tingkat penyembuhan. Namun untuk mendiagnosa kanker diperlukan alat screening seperti Computed Tomography (CT) Scan ataupun MRI sebelum dilakukan pengobatan. Di daerah terpencil dimana jumlah ahli atau radiologist berbanding terbalik dengan pasien, sehingga untuk melakukan manual checking hasil screening membutuhkan waktu yang lama. Selain itu, radiologist juga harus memeriksa gambar yang kompleks dengan tingkat abnomality yang beragam, sehingga kesalahan mungkin saja terjadi. Oleh karena itu, hal ini menyebabkan penggunaan komputer tidak dapat dihindari, dengan bantuan program [1] (CAD), dimana komputer dapat memproses dengan cepat dan tanpa lelah.

Dengan berkembangnya machine learning, program CAD pun ikut berkembang dengan pesat. Meskipun hasil dari CAD masih memerlukan pengecekan ulang oleh radiologist untuk memastikan ulang, program CAD sangatlah membantu memangkas waktu pemeriksaan. Kanker paru-paru adalah salah satu penyakit yang mematikan di seluruh dunia. Telatnya pendeteksian diagnosa menjadi salah satu faktor rendahnya tingkat kesembuhan penyakit ini. Untuk melakukan CAD dalam permasalahan kanker paru-paru diperlukan tiga langkah, yaitu pendeteksian gelembung paru-paru (nodules), lalu melakukan segmentasi pada nodule yang telah terdeteksi, dan proses selanjutnya adalah mengklasifikasi tekstur dari nodule tersebut. Ketiga langkah tersebut adalah fokus utama pada LNDB challenge yang dipublikasikan online.

Untuk mengatasi masalah tersebut, tugas akhir ini akan mengambil bagian tugas segmentasi saja. Dengan demikian hasil dari tugas akhir ini dapat dilanjutkan ke tahap selanjutnya. Tujuan pembuatan program ini dapat dilihat pada beberapa poin berikut:

- Menyegmentasi semua nodule yang diberikan ke program, nodule telah dipreproses sebelum menuju proses prediksi.
- Membandingkan kegunaan dari generator saat training dan berbagai tipe residual pada model training deep learning.

II. TINJAUAN PUSTAKA

Untuk mendukung program yang dibuat yaitu segmentasi nodule akan menggunakan referensi beberapa jurnal, seperti

Agung Pribadi, Departemen Informatika, Institut Sains dan Teknologi Terpadu Surabaya, Surabaya, Jawa Timur, Indonesia (e-mail: omegaudayana777@gmail.com)
Anang Kukuh Adisusilo, Departemen Informatika, Universitas Wijaya Kusuma Surabaya, Jawa Timur, Indonesia (e-mail: anang@uwks.ac.id).)

pada penelitian Fabian Isensee tahun 2019 tentang segmentasi tumor pada ginjal [2] dan pada penelitian Zongwei Zhou tahun 2018 tentang perbandingan kemampuan antara berbagai variasi U-Net untuk melakukan beberapa tugas segmentasi [3]. Pada jurnal pertama, peneliti menjelaskan bahwa kanker ginjal adalah masalah kesehatan serius dengan 400.000 kasus tiap tahun. Peneliti menyatakan bahwa semantic segmentation adalah langkah awal yang menjanjikan terhadap peningkatan hasil perawatan. Peneliti juga menyatakan bahwa sebagian mayoritas keberhasilan pada 3D image segmentation didasari oleh variasi dari arsitektur U-Net. Untuk itu peneliti mengembangkan dan mencoba melatih 3 tipe U-Net yaitu U-Net biasa tanpa residual, U-Net dengan residual dan pre-activation residual U-Net. Peneliti melatih model dengan stochastic gradient descent dan batch size dengan ukuran 2, juga mengaplikasikan data augmentation ke semua data training selama proses training berjalan. Data augmentasi yang diaplikasikan adalah scalling, rotation, brightness, gamma dan Gaussian noise. Berdasarkan pembahasan dan penelitian tersebut, maka dapat diambil kesimpulan bahwa data augmentasi dan berbagai macam tipe residual dapat dimanfaatkan untuk proses training.

Pada jurnal kedua, peneliti menyatakan bahwa skip connection yang menyambungkan encoder ke decoder telah terbukti ampuh menghasilkan segmentation mask dengan detail yang baik meskipun memiliki background yang kompleks, hal ini terlihat pada arsitektur U-Net biasa dan FCN yang memanfaatkan skip connection. Peneliti juga menyatakan bahwa menyegmentasi pada medical image menuntut akurasi tinggi karena jika terdapat error sedikit saja pada segmentasi dapat mengarah ke user experience yang rendah, sebagai contoh pola disekitar nodule paru-paru dapat mengindikasikan keganasan nodule. Oleh karena itu, diharapkan untuk adanya arsitektur image segmentation yang secara efektif memulihkan detail yang jelas pada objek target. Peneliti lalu merancang UNet++ yang memiliki arsitektur bersarang. Peneliti juga memiliki hipotesis bahwa model ini dapat secara efektif menangkap detail target secara jelas. Yang membedakan ini dari U-Net biasa yang secara langsung melempar hasil feature maps encoder ke decoder yang lalu menghasilkan perpaduan feature maps yang secara sementic tidak sama adalah pelemparan hasil feature maps pada encoder tidak langsung dilemparkan ke decoder tetapi melalui convolution block. Berdasarkan pembasahan dan penelitian tersebut, maka dapat diambil kesimpulan bahwa terdapat variasi lainnya selain U-Net yang dapat dicoba untuk training nantinya.

Untuk mencapai target, diperlukan beberapa komponen pendukung. Berikut adalah beberapa komponen pendukung untuk membuat program. Teori singkat diberikan agar pembaca dapat memahami secara singkat komponen tersebut.

A. Convolutional Neural Network

Convolutional Neural Network (CNN) adalah perkembangan dari multi level perceptron. CNN terinspirasi dari proses biologi pada pola konektivitas antara banyak neuron sehingga menyerupai visual cortex pada otak. CNN dapat dipakai pada input berdimensi 2 atau lebih, contohnya adalah gambar. Gambar jika diubah menjadi multi level perceptron (MLP) maka akan mendapat akurasi yang jelek. Oleh karena MLP mengabaikan spatial information dan memiliki banyak parameter yang redundant karena memakai fully connected layer sehingga tidak efisien.

$$P = (F_w * F_h * F_{n-1} + 1) * F_n \quad (1)$$

Pada formula 1, Parameter adalah weight yang dipelajari pada suatu layer. Penambahan angka 1 adalah karena pengaruh bias. Variabel $\boldsymbol{F_w}$ adalah ukuran lebar filter, variabel $\boldsymbol{F_w}$ adalah ukuran tinggi filter, variabel $\boldsymbol{F_{n-1}}$ adalah jumlah filter pada layer sebelumnya, variabel $\boldsymbol{F_n}$ adalah jumlah filter pada layer sekarang. Semakin besar total parameter sebuah model, maka semakin besar waktu yang diperlukan untuk melakukan training..

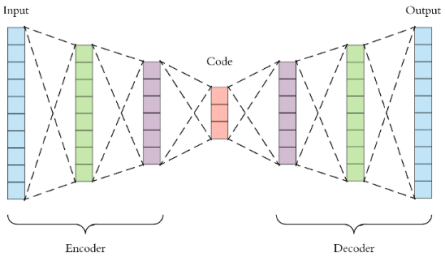
Karena gambar bergantung pada pixel, dan jika diubah menjadi 1 dimensi dan dilakukan proses dot matrix maka perhitungan akan kacau. Feed forward pada weight juga akan menjadi kacau karena posisi pixel, maka dari itu penggunaan multi perceptron jarang dipakai pada gambar. Layer convolutional memanfaatkan filter size untuk masalah ini.

Convolutional neural network teridiri dari layer convolutional dan layer-layer tambahan lainnya seperti pooling layer, activation function, layer upsampling, layer dropout[4], layer normalization[5] yang memiliki fungsi yang berbeda-beda. Layer-layer ini kemudian akan disusun dan membentuk model deep learning yang bisa dipakai untuk melatih tugas tertentu seperti klasifikasi.

B. Autoencoder

Autoencoder[6] adalah salah satu tipe neural network yang digunakan untuk feature learning dan biasanya bersifat unsupervised, dan dapat bersifat supervised juga. Feature learning didasari oleh keinginan agar tugas machine learning seperti klasifikasi menjadi lebih mudah, fitur dalam supervised dipelajari menggunakan input data yang sudah dilabeli, fitur dalam unsupervised dipelajari tanpa menggunakan label. Tujuan dari Autoencoder adalah mempelajari perwakilan atau fitur yang penting dari suatu data, biasanya seperti dimensionality reduction. Autoencoder berusaha menghasilkan data baru dari representasi data yang penting tersebut.

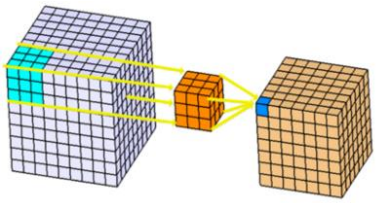
Autoencoder terdiri dari encoder dan decoder. Encoder berguna untuk mempelajari dan mengecilkan dimensi input. Decoder berguna untuk mempelajari dan memperbesar dimensi bottleneck hingga ke dimensi awal input. Hasil visual dari Autoencoder dapat lebih baik (seperti penggunaan denoising Autoencoder) atau lebih jelek (karena mengambil fitur yang penting saja). Autoencoder dapat bersifat stacked atau tidak, jika bersifat stacked maka neural network mempunyai banyak hidden layer, jika tidak maka neural network hanya memiliki 1 hidden layer.



Gambar 1. Struktur Autoencoder

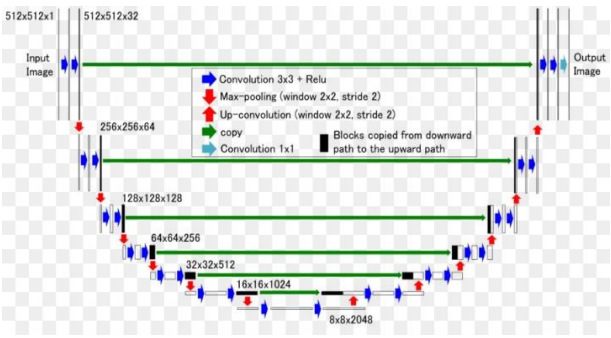
C. U-Net

Tujuan dari U-Net[7] adalah mapping fitur dari gambar (segmentasi), neural network ini dikhususkan untuk masalah klasifikasi untuk setiap pixel karena vector label untuk setiap pixelnya sudah disediakan oleh radiologist, dimana setiap file CT Scan bisa memiliki lebih dari 1 nodule, label dalam kasus ini adalah area nodule dengan nilai Id Finding yang dapat dilihat pada file csv. Target dari neural network adalah menghasilkan vector label. Unet memiliki encoder (down sampling) dan decoder (up sampling) yang sama seperti Autoencoder. Tiap layer encoder yang ada akan mengecilkan dimensi input layer tersebut, sehingga hal ini disebut down sampling. Tiap layer decoder yang ada akan membesarkan dimensi input layer tersebut, sehingga hal ini disebut up sampling. Hal yang membedakan kedua jenis adalah adanya layer penghubung encoder menuju decoder (layer concatenate). Unet biasanya menggunakan layer convolutional untuk bagian encoder dan decodernya ketimbang menggunakan layer Dense.



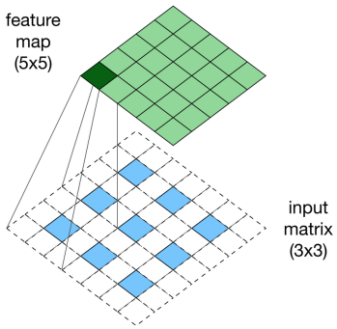
Gambar 2. Layer Convolutional 3 Dimensi

Unet dapat menerima input gambar 3 dimensi maupun 2 dimensi, dengan bantuan layer convolutional, hal ini menjadi mudah. Karena layer convolutional berupa 3 dimensi, maka weight juga berupa 3 dimensi. Proses perkalian input (bukan dot matrix) dengan weight akan menghasilkan output 3 dimensi juga. Dimensi output dapat diatur menggunakan stride dan padding, dimensi output dapat sama dengan dimensi input dengan bantuan padding dengan tipe same.
Hal yang membedakan U-Net dari Autoencoder adalah adanya layer skip connection yang menghubungkan hasil feature map sebagai input tambahan untuk layer decoder pada level yang sama dengan menggunakan bantuan layer concatenate, sehingga dimensi input akan bertambah, bukannya nilai dari input decoder yang bertambah.



Gambar 3. Struktur U-Net

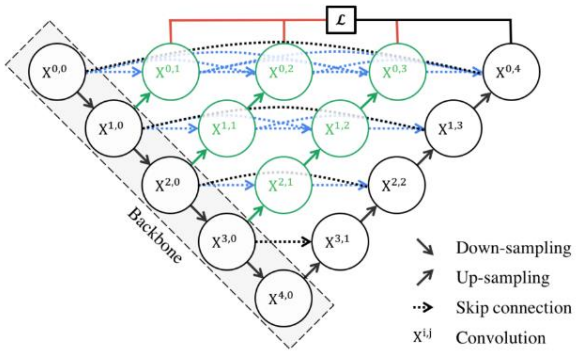
Pada gambar 3, garis hijau adalah layer concatenate yang menghubungkan encoder ke decoder. Garis merah ke bawah adalah layer downsampling yang biasanya menggunakan layer max pooling ataupun dapat memanfaatkan atribut stride pada layer convolutional. Garis merah ke atas adalah layer transpose. Transpose layer adalah teknik lain selain up sampling biasa. Transpose layer seringkali salah dikenal sebagai layer deconvolutional, padahal kedua layer adalah jenis yang berbeda. Layer deconvolutional memutar balikan operasi dari layer convolutional biasa. Sedangkan layer transposed meningkatkan spatial dimensi dari input dengan bantuan stride dan padding. Selain itu, layer transposed memiliki weight yang dapat dipelajari, cara mempelajari weight sama dengan layer convolutional pada umumnya, sehingga layer ini mempunya ukuran filter dan jumlah channel. Hal ini lah yang membedakan dari layer up sampling biasa. Hasil output dari layer deconvolutional dan layer transpose memiliki spatial dimension yang sama, sehingga hal ini yang membuat pemahaman kebanyakan orang salah.



Gambar 4. Visualisasi Layer Transpose

Pada gambar 4, input matrix awalnya adalah 3x3, tetapi karena terdapat padding 1 dan stride 2 maka akan keluar hasil seperti pada gambar. Kotak berwarna biru adalah kotak dengan nilai input awal matrix, sedangkan kotak tidak berwarna memiliki nilai 0. Langkah selanjutnya adalah dilakukan perkalian dengan weight seperti layer convolutional. Jika stride adalah 3 dan padding bertipe same, maka dimensi input awal akan dikalikan dengan 3.

D. U-Net++



Gambar 5. Struktur Unet++

Struktur Unet++ terdiri dari Unet sebagai bagian utama encoder dan decodernya (backbone dan hasil upsampling tiap decoder). Layer diantara encoder dan decoder utama adalah layer hasil concenate layer upsampling layer convolutional level x+1 dengan layer convolutional yang ada pada level x

(level dimulai dari paling atas), layer ini ditandai dengan warna hijau pada gambar 5. Selain itu, layer hijau juga menerima concatenate dari layer-layer sebelumnya pada level yang sama. Sebagai contoh layer X0,3 pada gambar adalah hasil up sampling dari layer X1,2 dan menerima concatenate dari X0,0 dan X0,1 dan X0,2. Setelah itu biasanya hasil tersebut akan di feed forward ke layer convolutional. Hal ini untuk memberikan input ke layer-layer setelahnya sehingga dapat di-feed forward untuk mendapatkan informasi gabungan. Jumlah filter pada layer convolutional untuk setiap level memiliki jumlah yang sama dan spatial dimension output yang sama (selain hasil concatenate dan upsampling).

III. SEGMENTASI NODULE

Pada bab ini akan dijabarkan, yaitu mengenai dataset, arsitektur program dan struktur model.

A. Dataset

Tersedia 236 file gambar dan 1527 baris csv yang dapat dijadikan menjadi dataset gambar, dengan atribut id file yang dipakai, id radiologis, id nodule yang ada pada 1 gambar, lokasi centroid x, lokasi centroid y, lokasi centroid z, apakah ini nodule, volume nodule dan tekstur nodule. Pada kasus segmentasi, atribut tekstur nodule dan apakah ini nodule tidak digunakan. File-file tersebut didapat dari LNDb challenge[8] yang dimulai pada tahun 2019. Setelah proses prediksi segmentasi berhasil dilakukan, dapat dilanjutkan proses klasifikasi tesktur nodule mulai dari nilai 0 sampai 5, dimana 0 adalah bukan nodule. Tetapi, proses klasifikasi tekstur tidak dilakukan karena bukan fokus utama pengerjaan.

LNDbID	RadID	FindingID	x	y	z	Nodule	Volume	Text
1	1	1	-44.6084	-119.073	-37.5	1	440.9088	5
1	1	2	25.85254	-126.97	-45.5	1	152.381	4
1	2	1	-44.001	-118.466	-37.5	1	56.82005	5
1	3	1	-44.001	-119.681	-37.5	1	169.3533	5
2	1	1	88.89551	-123.626	-129.5	1	339.188	5
2	1	2	63.53418	-112.757	-117.5	1	163.2933	5
2	1	3	-103.851	-117.104	-253.5	1	357.0399	5
2	1	4	67.88184	-95.3662	-81.5	1	54.08105	3
2	1	5	44.69434	-123.626	-63.5	1	22.05247	4
2	2	1	88.89551	-124.351	-129.5	1	282.4816	5

Gambar 6. Gambaran Isi File Dataset

Pada gambar 6, terdapat nodule dengan titik centroid x,y dan z yang sama yang telah disegment oleh lebih dari 1 radiologist. Setiap radiologist memberikan hasil segmentasi menurut pendapat mereka masing-masing, sehingga hasil tersebut tidak sama satu sama lain. Hal ini dapat dimanfaatkan untuk melihat distribusi nodule pada dataset utama sebelum di split untuk training, validasi dan testing.

LNDbID	RadID	RadFinding	FindingID	x	y	z	AgrLevel	Nodule	Volume	Text
1	1,2,3	1,1,1	1	-44.2035	-119.073	-37.5	3	1	222.3607	5
1	1	2	2	25.85254	-126.97	-45.5	1	1	152.381	4
2	1,2,3	1,1,3	1	88.89551	-123.868	-129.5	3	1	378.0423	5
2	1,3	2,2	2	63.53418	-112.757	-117.5	2	1	174.8446	5
2	1,3	3,5	3	-103.851	-116.742	-253	2	1	297.7083	5
2	1	4	4	67.88184	-95.3662	-81.5	1	1	54.08105	3
2	1	5	5	44.69434	-123.626	-63.5	1	1	22.05247	4
2	2	2	6	117.8799	-121.452	-209.5	1	1	14.13717	5
2	3	1	7	87.44629	-121.452	-107.5	1	1	384.8681	4

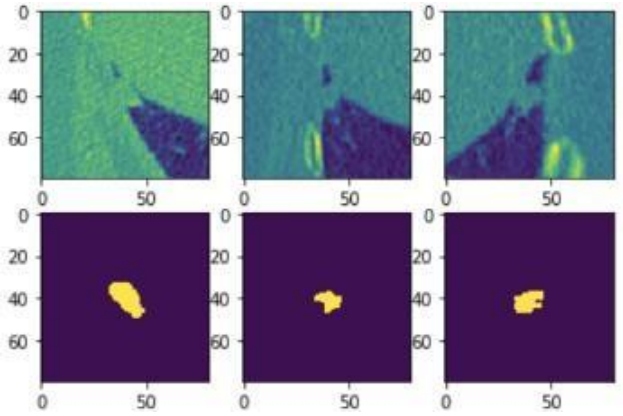
Gambar 7. File Kumpulan Nodule yang Sama

Pada gambar 7, informasi tentang nodule yang sama telah diringkas sehingga dapat dengan mudah dibuat distribusi data nya. Nilai seperti x, y, z, volume dan texture adalah nilai rata-rata dari kumpulan data awal. Nilai rata-rata pada informasi di file ini tidak dipakai, program akan memakai nilai x, y, z awal pada gambar 6. Dari file csv gambar 7, akan dapat sebuah kesimpulan yaitu, jumlah nodule dengan frekuensi 0 adalah 594, dengan frekuensi 1 adalah 432, dengan frekuensi 2 adalah 151, dengan frekuensi 3 adalah 42.

1528 data akan dipangkas menjadi 860 data karena perhitungan volume nodule yang akan diambil adalah dengan lebar diameter lebih besar sama dengan 3 mm. Angka 3 mm dipilih karena angka dibawah itu adalah micronodule, micronodule dapat berevolusi menjadi nodule dan jika micronodule terus membesar diameternya maka dapat bersifat (ada kemungkinan) berbahaya dan dapat bersifat kanker. Perhitungan diameter nodule menggunakan rumus volume sphere.

$$V = \frac{4}{3}\pi r^3 \quad (2)$$

Pada dataset, volume sphere sudah diketahui maka variabel yang harus dicari adalah r (radius atau jari-jari), setelah menemukan jari-jari, maka langkah selanjutnya adalah hasil jari-jari dikalikan dengan 2 agar menghasilkan diameter sphere. Pembagian data adalah 70% untuk training, 20% untuk validasi, 10% untuk testing.



Gambar 8. Contoh Nodule dan Segmentasinya

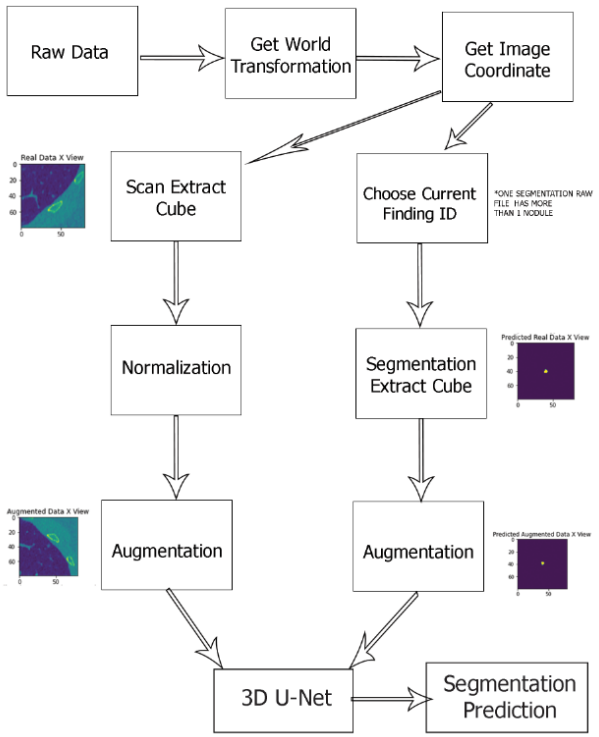
Pada gambar 8, gambar sudah dilakukan preprocessing dan hasil outputnya berupa kubus dengan dimensi 80x80x80 pixel. Dapat terlihat sebuah objek berbentuk menyerupai gelembung. Posisi gelembung untuk semua dataset pasti berada pada posisi tengah karena pengaruh titik centroid x, y dan z yang ada di dataset.

B. Arsitektur Program

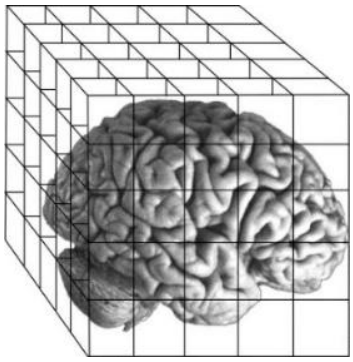
Program akan mengambil raw data dari dataset menggunakan library simpleITK sesuai dengan informasi dari file csv. Hasil rawdata akan diubah menjadi tipe numpy sehingga mudah diolah oleh library lainnya. Setelah itu, titik-titik centroid akan digunakan untuk membuat area kubus untuk nodule yang ditunjuk dan juga dilakukan hal yang sama untuk target segmentasi nodule. Setelah itu kubus nodule dan segmentasinya akan menjadi input untuk model training.

Input dari program ini adalah raw data. Library simpleITK membantu pengolahan raw data sehingga dapat dibaca, seperti mendapatkan nilai spacing, origin dan rotation matrix. Variabel-variabel tersebut sangat penting untuk mengubah sistem koordinasi CT scan menjadi sistem koordinasi gambar biasa. Raw data memiliki Anatomical

Coordinate System, simpleITK sudah mengubah raw data menjadi Image Coordinate System, tetapi titik centroid pada file csv belum dirubah menjadi Image Coordinate System.



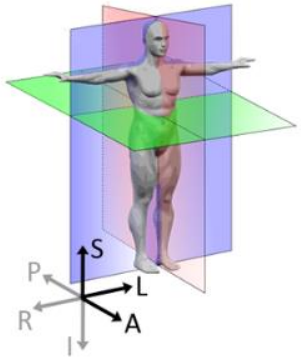
Gambar 9. Arsitektur Program



Gambar 10. Contoh Voxel Pada Gambar 3 Dimensi

Voxel adalah volumetric pixel penyusun gambar 3 dimensi. Voxel adalah panggilan bongkahan pixel dalam gambar 3 dimensi, dalam 1 voxel terkandung informasi yang tidak dimiliki oleh pixel, karena voxel memiliki banyak pixel, sedangkan pixel hanya memiliki 1 pixel itu sendiri. Voxel tidak harus berbentuk kubus, namun bisa berbentuk segitiga, diamond, bola, maupun bidang lainnya yang dapat ditumpuk menghasilkan voxel.

Untuk pengerjaan tugas ini, sistem koordinat dibagi menjadi 3 yaitu world coordinate system, anatomical coordinate system dan image coordinate system[9]. World coordinate system adalah sistem koordinat kartesian dimana kita ingin menentukan dimana gambar ditampilkan, posisi originnya biasanya berada di pojok kiri bawah, tetapi nilai originnya dapat dirubah.

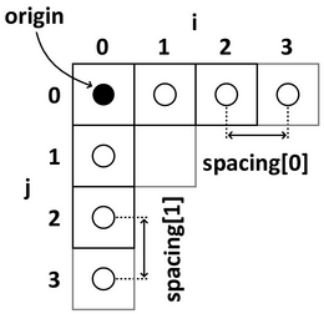


Gambar 11. Contoh Anatomical Coordinate System

Anatomical Coordinate System adalah sistem koordinat yang dipakai untuk teknik pencitraan medis. Sistem ini terdiri dari 3 bidang, yaitu:

- Bidang axial yang berdiri tegak dari tanah menuju kepala. Pada gambar 11, bidang ini diwakilkan oleh variabel S (Superior) dan I (Inferior).
- Bidang coronal yang memisahkan antara depan dan belakang. Pada gambar 11, bidang ini diwakilkan oleh variabel A (Anterior) dan P (Posterior).
- Bidang sagittal yang membentang dari kiri hingga kanan. Pada gambar 11, bidang ini diwakilkan oleh variabel L (Left) dan R (Right).

Image coordinate system menjelaskan bagaimana nilai pixel dalam gambar diletakan di dalam gambar. Format foto JPEG atau format lainnya mempunyai standard untuk pengkoordinasian. Selain itu, sistem ini juga menjelaskan orientasi (landscape atau portrait). Perbedaan antara image coordinate dengan world coordinate adalah origin, image coordinate tidak dipengaruhi oleh origin, sedangkan world coordinate dipengaruhi



Gambar 12. Gambar Penjelasan Spacing dan Origin

Spacing adalah jarak antara titik centroid voxel dengan titik centroid voxel lainnya. Origin adalah posisi awal voxel pertama, biasanya berada di pojok kiri atas. Sumbu i akan bergerak ke kanan dan sumbu j akan bergerak ke bawah. Setiap gambar raw memiliki 1 buah nilai origin, spacing dan rotation matrix

$$R = \begin{bmatrix} cos\,\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \quad (3)$$

Pada formula 3, Rotation matrix biasanya berupa matrix identitas. Matrix ini mengatur apakah titik koordinat voxel memiliki rotasi dari raw data. Nilai Θ biasanya adalah 0

karena tidak terjadi rotasi, jika nilai 0 maka secara matematis matrix akan menjadi matrix identitas. Input masih memiliki sistem perkoordinasian anatomical, sehingga langkah selanjutnya adalah mengubahnya menjadi image coordinate system dan mengekstrak daerah sekitar nodule sehingga input deep learning tidak perlu menerima banyak informasi tidak penting, seperti mempelajari semua area CT Scan.

Preprocessing diperlukan agar input data dapat diringkas sehingga dapat digunakan untuk deep learning. Sistem koordinat pada titik centroid x, y, z yang ada pada dataset perlu dirubah menjadi Image Coordinate System. Berikut adalah penjelasan dan langkah-langkah preprocessing:

- Mendapatkan World Transformation

Setelah mendapatkan nilai variabel origin, spacing dan rotation matrix maka langkah selanjutnya adalah mengubah sistem koordinasi dataset menuju ke world coordinate system untuk mendapatkan transformasi matrixnya, karena jika transformasi matrix milik world coordinate di inverse maka akan menjadi transformasi matrix image coordinate.

$$\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix} = \begin{bmatrix} A_{11} * S_1 & A_{12} * S_2 & A_{13} * S_3 \\ A_{21} * S_1 & A_{22} * S_2 & A_{23} * S_3 \\ A_{31} * S_1 & A_{32} * S_2 & A_{23} * S_3 \end{bmatrix} \quad (4)$$

Pada formula 3, variabel A adalah rotation matrix dan variabel S adalah nilai spacing untuk tiap sumbu atau axis. Rotation matrix dikalikan dengan spacing karena pada world coordination system dan image coordination system tidak menyimpan nilai spacing. Langkah selanjutnya adalah menginverse transformasi matrix yang didapatkan.

- Mengubah Titik Centroid Menjadi Image Coordinate System

Titik-titik centroid yang terdapat pada file csv perlu diubah sistem koordinatnya. Hal ini karena gambar raw sudah berubah menjadi Image Coordinate System (bertipe numpy) setelah dibaca oleh library simpleITK. Titik centroid ini akan memberitahu lokasi titik tengah dari nodule, setelah diubah sistem koordinatnya, maka crop daerah sekitar nodule akan menjadi mudah.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} \quad (5)$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{bmatrix}^{-1} \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \quad (6)$$

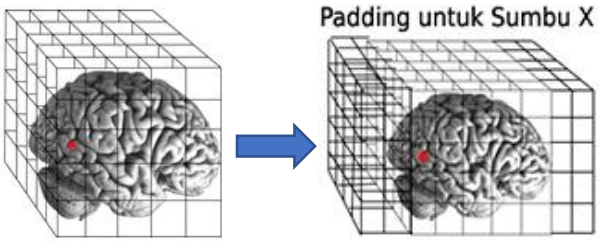
Pada formula 5 dan 6 digunakan untuk mengubah posisi menjadi koordinat Image Coordinate System, variabel T adalah nilai origin untuk masing-masing sumbu atau axis. Image Coordinate System tidak memiliki nilai origin sehingga, titik-titik koordinat dikurangi dengan nilai origin. Setelah dikurangi, hasil tersebut dikalikan matriks dengan transformasi matriks World Coordinate System yang sudah dilakukan inverse. Setelah mengubah sistem koordinat, langkah selanjutnya adalah mengambil daerah sekitar nodule.

- Ekstrak Kubus

Langkah pertama untuk mengekstrak daerah sekitar nodule adalah menentukan ukuran untuk 1 voxel, karena voxel pada dataset adalah kubus maka 1 ukuran untuk semua sisi, dan menentukan dimensi output yang diinginkan. Semakin besar ukuran voxel maka gambar nodule yang ditampilkan akan semakin kecil, karena 1 voxel yang lebih besar menampung lebih banyak informasi gambar, dalam kasus ini nodule ada pada titik tengah voxel.

$$HC_i = \frac{1}{2} * S_i * L \quad (7)$$

Pada formula 7, rumus tersebut untuk mendapatkan setengah bagian dari voxel atau cube untuk masing-masing axis atau sumbu. Rumus tersebut akan menghasilkan 3 hasil untuk masing-masing axis. Hal ini dilakukan agar nodule bisa berada di tengah voxel. Variable Si adalah nilai spacing untuk axis yang ke i. Variabel L adalah nilai ukuran untuk 1 voxel. Setelah mendapatkan ukuran setengah voxel, langkah selanjutnya adalah mengecek apakah posisi centroid voxel jika ditambah atau dikurangi nilai setengah apakah melewati batas resolusi gambar.



Gambar 13.. Ilustrasi Padding untuk Sumbu X

Jika melewati maka akan dilakukan padding untuk semua sumbu, padding dengan nilai 0 dilakukan sebanyak nilai maximum dari nilai Hc untuk sisi kiri, bawah dan depan, nilai maximum juga untuk sisi kanan, atas dan belakang. Karena pengaruh dari padding padding maka titik centroid yang sudah diubah menjadi Image Coordinate System ditambah dengan nilai maximum tersebut (lihat posisi titik merah pada gambar, gambar sudah menjadi Image Coordinate System sejak dibaca dengan library simpleITK).

Jika tidak melewati dimensi gambar, maka dilakukan pemotongan gambar dimulai dari titik centroid x-Hcx sampai x + Hcx, dari titik centroid y-Hcy sampai y-Hcy, dari titik centroid z sampai z + Hcz. Hasil pemotongan bisa saja tidak sesuai dengan dimensi output yang diinginkan(80x80x80), maka dilakukan proses zoom sengan skala dibagi dengan dimensi hasil x untuk proses zoom sumbu x, sumbu lain pun dilakukan proses yang sama.

- Memilih Finding Id pada Data Segmentasi

Masing-masing CT Scan memiliki informasi segmentasinya sendiri-sendiri yang berada pada file yang berbeda dengan file CT Scan. Setelah data segmentasi telah melalui proses ekstrak cube dengan dimensi 80x80x80. Dalam informasi kubus tersebut dapat memiliki lebih dari 1 nodule, hal ini ditandai dengan angka. Seperti angka 0 menandakan bukan area nodule, angka 1 dan seterusnya adalah id nodule pada kubus tersebut. Id finding dapat dilihat

pada file csv dataset, selain id finding pada iterasi pembacaan file akan dijadikan nilai 0 dan nilai id finding pada kubus akan dijadikan angka 1.

- Normalisasi

Tipe dari gambar biasanya adalah RGB (Red, Green, Blue), yang terdiri dari nilai 0 sampai 255. Namun, tipe pada dataset ini adalah tipe Hounsfield Unit (HU) yang terdiri dari nilai antara -1000 sampai 4000. Nilai-nilai tersebut menentukan bagian dari organ tubuh seperti tulang memiliki nilai sekitar 100 HU sampai 2200 HU dan lemak ditandai dengan nilai sekitar -120 sampai -90.

Normalisasi dibutuhkan agar saat training jika misal value pada dataset ke 1 index x adalah 100 dan dataset ke 2 index x adalah 3, maka perubahan weight saat backpropagation akan sangat besar. Oleh karena itu, dibutuhkan normalisasi data input, meskipun value pada index x berbeda, namum tidak akan mengakibatkan perubahan weight yang sangat besar. Normalisasi yang dipakai pada dataset adalah normalisasi min max, dengan memanfaatkan range nilai min dan max pada 1 data dapat digunakan untuk melakukan normalisasi. Berikut adalah rumus normalisasi min max.

$$\boldsymbol{t_{range} = t_{max} - t_{min}} \quad (8)$$

$$\boldsymbol{s_{range} = s_{max} - s_{min}}$$

$$\boldsymbol{scale} = \frac{(\boldsymbol{array_{image} - s_{min}})}{\boldsymbol{s_{range}}}$$

$$\boldsymbol{result} = \boldsymbol{t_{min}} + (\boldsymbol{scale} * \boldsymbol{t_{range}})$$

Pada formula 8, target normalisasi adalah 0 untuk nilai minimumnya dan 1 untuk nilai maximumnya. Variabel t adalah target, s adalah sumber, sehingga $\boldsymbol{s_{max}}$ dan $\boldsymbol{s_{min}}$ ditentukan sendiri nilainya. Rumus diatas dapat bekerja jika keempat parameter sudah diketahui yaitu: nilai maximum target, nilai minimum target, nilai maximum sumber image, nilai minimum sumber image. Rumus ini juga dapat dipakai untuk denormalisasi, yaitu mengembalikan nilai ke asalnya jika mengetahui keempat parameternya. Variabel $\boldsymbol{array_{image}}$ adalah sebuah array yang mengandung informasi gambar. Variabel $\boldsymbol{scale}$ akan menghasilkan output sama dengan dimensi gambar.

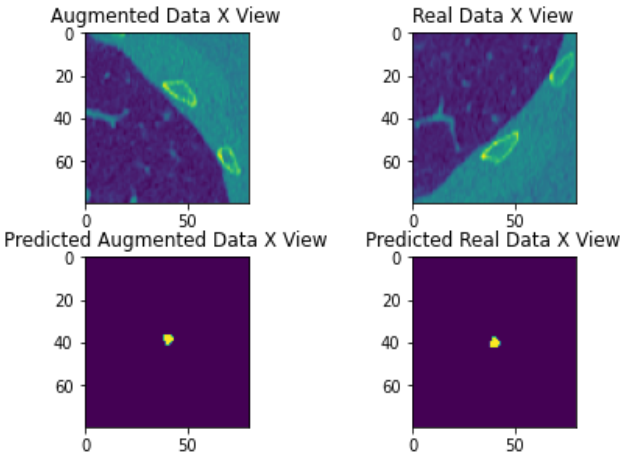
Pada saat pengerjaan, normalisasi dilakukan untuk masing-masing gambar, sehingga nilai maximum dan minimum sumber akan berbeda-beda seperti karena data CT scan tidak semuanya terdapat udara yang nilainya HU nya -1000 atau nilai HU lainnya. Normalisasi ini tidak mengikuti nilai maximum dan minimum Hounsfield Unit yaitu -1000 untuk minimum dan 4000 untuk maximum. Hal ini dilakukan untuk mengacak nilai (hasil normalisasi) nodule pada gambar untuk deep learning.

- Data Augmentation

Data augmentation digunakan untuk menambah variasi dari nodule. Karena nodule selalu berada di tengah maka augmentasi yang akan dilakukan adalah rotasi, flip dan brightness. Akan ada 400 data augmentasi tambahan untuk dataset utama. Beberapa augmentasi yang digunakan adalah sebagai berikut.

1. Rotasi Gambar

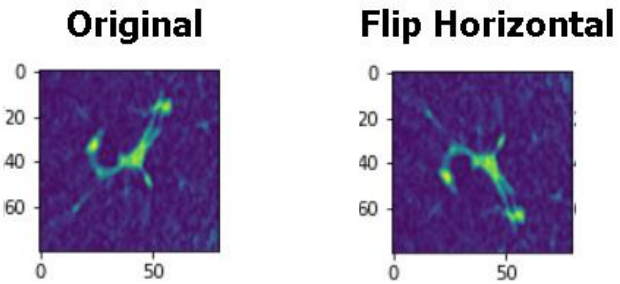
Proses ini akan memutar gambar secara 3 dimensi dengan titik centroid kubus sebagai pusat perputaran. Sudut perputaran dipilih secara acak dengan range antara 30 derajat hingga -30 derajat. Rotasi gambar dapat dilakukan dengan mudah dengan bantuan library Scipy. Contoh gambar yang telah diaplikasikan dengan rotasi dapat dilihat pada gambar 14.



Gambar 14 Contoh Augmentasi Rotate

2. Membalikkan Gambar

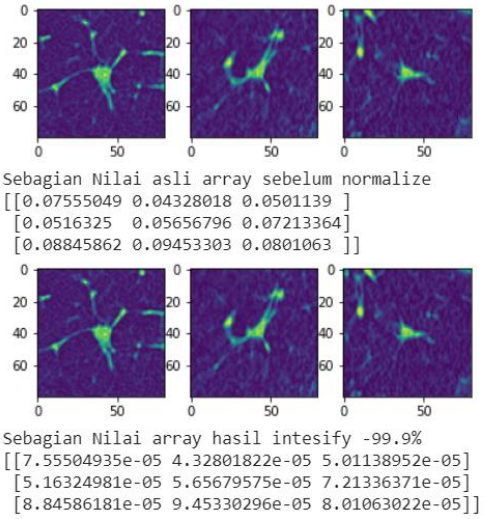
Proses ini akan membalikkan gambar secara horizontal ataupun vertical. Pemilihan sumbu tersebut akan dipilih secara acak. Proses ini dapat dengan mudah dilakukan juga dengan menggunakan library Scipy. Contoh gambar yang telah diaplikasikan dengan membalikkan gambar dapat dilihat pada gambar 15.



Gambar 15. Contoh Augmentasi Flip Horizontal

3. Intensitas Pixel

Proses ini akan mengurangi nilai setiap pixel sebanyak nilai persen yang ditentukan. Proses ini dapat diselesaikan tanpa menggunakan library apapun, hanya menggunakan perkalian biasa. Meskipun nilai pixel telah berubah drastis, terang gelapnya tidak dapat dilihat pada gambar. Pengubahan intensitas gambar pada program berada di range antara 10% hingga -10%. Contoh gambar yang telah diaplikasikan dapat dilihat pada gambar 16.

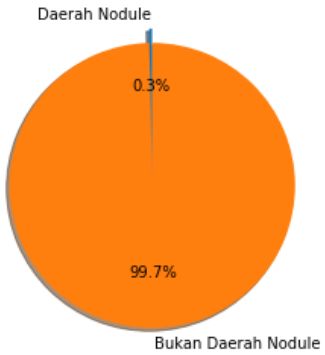


Gambar 16. Contoh Augmentasi Intensity.

Dengan adanya data augmentasi maka distribusi data akan berubah menjadi: frekuensi 0 adalah 594, dengan frekuensi 1 adalah 234, dengan frekuensi 2 adalah 242, frekuensi 3 adalah 79, dengan frekuensi 4 adalah 48, dengan frekuensi 5 adalah 19, dengan frekuensi 6 adalah 3.

- Training

Proses training menerima input dataset yang sudah melewati preprocessing dan data hasil augmentasi, juga menerima target segmentasi dataset asli dan segmentasi hsail augmentasi. Setiap dataset pasti memiliki nodule. Training dilakukan dengan menggunakan dice loss, penggunaan loss ini karena kelas untuk nodule dan bukan nodule sangat tidak seimbang.



Gambar 17. Distribusi Nilai Target

Pada gambar 17, data distribusi berasal dari input training yang berjumlah 70% dari 1260 gambar (860 gambar original, 400 gambar augmentasi). Terlihat bahwa distribusi dikuasai oleh daerah bukan nodule atau nilai 0 pada target sebanyak 99.7%, dan distribusi daerah nodule atau nilai 1 pada target sangat kecil sebanyak 0.3%. Jika menggunakan loss function binary crossentropy, hasil prediksi akan jelek karena pengaruh nilai 0 pada target.

$$\boldsymbol{Loss\ Function} = \mathbf{1} - \frac{\mathbf{2}\sum_{i=0}^{h}\sum_{j=0}^{w}(p_{ij} * \widehat{p_{ij}})}{\sum_{i=0}^{h}\sum_{j=0}^{w}(p_{ij} + \widehat{p_{ij}})} \quad (9)$$

Pada formula 9, variabel $\widehat{p_{ij}}$ adalah nilai prediksi model dan variabel p_{ij} adalah nilai target. terjadi proses perkalian pada pecahan pembilang. Hal ini sangat penting karena dengan mengkalikan, maka jika salah satu memiliki nilai 0 maka nilai Dice Loss tidak akan terpengaruh. Pada bagian penyebut pecahan pun nilai 0 tidak memiliki efek apapun yang mempengaruhi nilai dice loss. Dice coefficient pada training akan digunakan sebagai metric.

- **Testing**

Pengukuran akurasi testing akan menggunakan Jaccard Index. Daerah interseksi akan dimanfaatkan untuk mengukur kesamaan antara data testing dan memiliki rumus daerah interseksi dibagi dengan union dari data. Formula ini cocok untuk membandingkan kesamaan antara data prediksi dengan data target.

$$\boldsymbol{J(A,B)} = \frac{|\boldsymbol{A} \cap \boldsymbol{B}|}{|\boldsymbol{A}| + |\boldsymbol{B}| - |\boldsymbol{A} \cap \boldsymbol{B}|} \quad (10)$$

Pada formula 10, variabel A adalah data target variabel B adalah data prediksi. Nilai dari interseksi akan dibagi dengan total penjumlahan area noduleA dengan area nodule B lalu dikurangi dengan daerah interseksi. Formula seperti ini juga dikenal sebagai Intersection Over Union (IoU).

C. Struktur Model

Struktur model akan ditulis dengan bantuan block. Block terhubung dengan struktur utama. Output dimensi block ditampilkan pada struktur model utama. Sebuah layer terhubung dengan layer yang ada pada kolom ‘Connected To’. Dimensi input dan output memiliki nilai yang sama karena menggunakan tipe padding same, kecuali layer shortcut projection, max pooling dan transpose.

- Model U-Net (Dua Conv32-Instnorm-ReLU Max Pooling Dice Loss)

Berikut adalah struktur model pertama:

TABEL 1
STRUKTUR BLOCK MODEL U-NET

Block	
Layer	**Connected To**
conv3dBlock (Conv3d, ?, 3x3x3)	inputBlock
instancenormalizationBlock (InstanceNormalization)	conv3dBlock
reluBlock	instancenormalizationBlock

Pada tabel 1, simbol ‘?’ merupakan parameter untuk banyaknya layer convolutional. Nilai ini didapat dari parameter struktur layer di model utama. Setiap layer convolutional pada block memiliki convolutional size 3x3x3 dan stride 1x1x1. Model ini tidak memiliki residual, yang

membedakan stuktur block dengan model E adalah jumlah stack atau tumpukan block.

TABEL 2
STRUKTUR MODEL U-NET

Layer	Output	Connected To
input1(InputLayer)	[(80,80,80,1)]	-
Block1 (32)	(80,80,80,32)	input1
Block2 (32)	(80,80,80,32)	Block1
maxpooling3d(MaxPooling3D)	(40,40,40,32)	Block2
Block3 (64)	(40,40,40,64)	maxpooling3d
Block4 (64)	(40,40,40,64)	Block3
maxpooling3d1(MaxPooling3D)	(20,20,20,64)	Block4
Block5 (128)	(20,20,20,128)	maxpooling3d1
Block6 (128)	(20,20,20,128)	Block5
maxpooling3d2(MaxPooling3D)	(10,10,10,128)	Block6
Block7 (256)	(10,10,10,256)	Maxpooling3d2
Block8 (256)	(10,10,10,256)	Block7
maxpooling3d3(MaxPooling3D)	(5,5,5,256)	Block8
Block9 (512)	(5,5,5,512)	maxpooling3d3
Block10 (512)	(5,5,5,512)	Block9
conv3dtranspose(Conv3DTranspose)	(10,10,10,256)	Block10
concatenate(Concatenate)	(10,10,10,512)	conv3dtranspose Block8
Block11 (256)	(10,10,10,256)	concatenate
Block12 (256)	(10,10,10,256)	Block11
conv3dtranspose1(Conv3DTranspose)	(20,20,20,128)	Block12
concatenate1(Concatenate)	(20,20,20,256)	conv3dtranspose1 Block6
Block13 (128)	(20,20,20,128)	concatenate1
Block14 (128)	(20,20,20,128)	Block13
conv3dtranspose2(Conv3DTranspose)	(40,40,40,64)	Block14
concatenate2(Concatenate)	(40,40,40,128)	conv3dtranspose2 Block4
Block15 (64)	(40,40,40,64)	concatenate2
Block16 (64)	(40,40,40,64)	Block15

TABEL 2 (LANJUTAN)
STRUKTUR MODEL U-NET

Layer	Output	Connected To
conv3dtranspose3(Conv3DTranspose)	(80,80,80,32)	Block16
concatenate3(Concatenate)	(80,80,80,64)	conv3dtranspose3 Block2
Block17 (32)	(80,80,80,32)	concatenate3
Block18 (32)	(80,80,80,32)	Block17
conv3d(Conv3D, 1, 1x1x1)	(80,80,80,1)	Block18

Penumpukan block pada model ini adalah 2. Model ini memiliki 22,581,217 parameter sehingga membutuhkan waktu yang lama untuk training dan memory yang besar. Penumpukan block dilakukan karena dengan penumpukan blok jenis ini memiliki akurasi yang tinggi, sehingga ingin dilihat apakah dengan penumpukan 2 block akan menghasilkan akurasi yang lebih tinggi atau tidak.

- Model U-Net++ (Uplus Dua Conv32-Instnorm-ReLU Max Pooling Dice Loss)

Berikut adalah struktur model kedua:

TABEL 3
STRUKTUR RESIDUAL BLOCK MODEL U-NET++

ResidualBlock	
Layer	**Connected To**
conv3dBlock(Conv3d, ?, 3x3x3)	inputBlock
instancenormalizationBlock (InstanceNormalization)	conv3dBlock
reluBlock (ReLU)	instancenormalizationBlock
conv3dBlock1(Conv3d, ?, 3x3x3)	reluBlock
instancenormalizationBlock1 (InstanceNormalization)	conv3dBlock1
reluBlock1 (ReLU)	instancenormalizationBlock1

Pada tabel 3. Setiap layer convolutional pada block memiliki convolutional size 3x3x3 dan stride 1x1x1. Model ini tidak memiliki residual. Hal yang membedakan stuktur block dengan model G adalah layer activation dan layer normalisasi.

TABEL 4
STRUKTUR MODEL U-NET++

Layer	Output	Connected To
input1(InputLayer)	[(80,80,80,1)]	-
Block_1 (32)	(80,80,80,32)	input1

TABEL 4 (LANJUTAN)
Struktur model U-Net++

Layer	Output	Connected To
maxpooling3d(MaxPooling3D)	(40,40,40,32)	Block_1
Block_2 (64)	(40,40,40,64)	maxpooling3d
maxpooling3d1(MaxPooling3D)	(20,20,20,64)	Block_2
Block_3 (128)	(20,20,20,128)	maxpooling3d1
maxpooling3d2(MaxPooling3D)	(10,10,10,128)	Block_3
Block_4 (256)	(10,10,10,256)	maxpooling3d2
maxpooling3d3(MaxPooling3D)	(5,5,5,256)	Block_4
Block_5 (512)	(5,5,5,512)	maxpooling3d3
conv3dtranspose(Conv3DTranspose)	(10,10,10,256)	Block_5
concatenate(Concatenate)	(10,10,10,512)	conv3dtranspose Block_4
Block_6 (256)	(10,10,10,256)	Concatenate
conv3dtranspose1(Conv3DTranspose)	(20,20,20,128)	Block_4
Concatenate1(Concatenate)	(20,20,20,256)	conv3dtranspos1 Block_3
Block_7 (128)	(20,20,20,128)	Concatenate1
conv3dtranspose2(Conv3DTranspose)	(20,20,20,128)	Block_6
Concatenate2(Concatenate)	(20,20,20,384)	conv3dtranspos2 Block_3 Block_7
Block_8 (128)	(20,20,20,128)	Concatenate2
conv3dtranspose3(Conv3DTranspose)	(40,40,40,64)	Block_3
Concatenate3(Concatenate)	(40,40,40,128)	conv3dtranspos3 Block_2
Block_9 (64)	(40,40,40,64)	Concatenate3
conv3dtranspose4(Conv3DTranspose)	(40,40,40,64)	Block_7
Concatenate4(Concatenate)	(40,40,40,192)	conv3dtranspos4 Block_2 Block_9
Block_10 (64)	(40,40,40,64)	Concatenate4
conv3dtranspose5(Conv3DTranspose)	(40,40,40,64)	Block_8
Concatenate5(Concatenate)	(40,40,40,256)	conv3dtranspos5 Block_2 Block_9 Block_10
Block_11 (64)	(40,40,40,64)	Concatenate5

TABEL 4 (LANJUTAN)
STRUKTUR MODEL U-NET++

Layer	Output	Connected To
conv3dtranspose6(Conv3DTranspose)	(80,80,80,32)	Block_2
Concatenate6(Concatenate)	(80,80,80,64)	conv3dtranspos6 Block_1
Block_12 (32)	(80,80,80,32)	Concatenate6
conv3dtranspose7(Conv3DTranspose)	(80,80,80,32)	Block_9
Concatenate7(Concatenate)	(80,80,80,96)	conv3dtranspos7 Block_1 Block_12
Block_13 (32)	(80,80,80,32)	Concatenate7
conv3dtranspose8(Conv3DTranspose)	(80,80,80,32)	Block_10
Concatenate8(Concatenate)	(80,80,80,128)	conv3dtranspos8 Block_1 Block_12 Block_13
Block_14 (32)	(80,80,80,32)	Concatenate8
conv3dtranspose9(Conv3DTranspose)	(80,80,80,32)	Block_11
Concatenate9(Concatenate)	(80,80,80,128)	conv3dtranspos9 Block_1 Block_12 Block_13 Block_14
Block_15 (32)	(80,80,80,32)	Concatenate9
conv3(Conv3d, 1, 1x1x1)	(80,80,80,1)	Block_15

Penumpukan layer convolutional beserta activationnya pada layer ini adalah 2 (terlihat pada tabel 3). Layer ini memiliki 30,563,713 parameter sehingga membutuhkan waktu dan memory yang besar. Pada model ini ingin diketahui apa tidak perbedaan arsitektur biasa dengan arsitektur U-Net++.. Unet++ berusaha agar informasi dari layer encoder tidak hilang hingga sampai ke decoder. Sehingga dibuat banyak layer baru diantara encoder dan decoder pada kedalaman yang sama untuk menampung dan memproses informasi tersebut dengan bantuan concatenate.

IV. UJI COBA

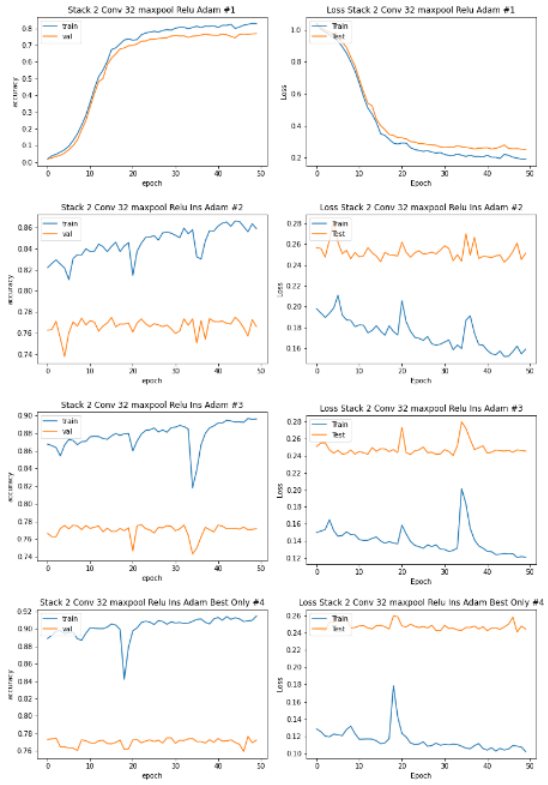
Uji coba dibagi menjadi 2 bagian yaitu model unet dan model Unet++. Saat melakukan sesi training akan dipecah menjadi beberapa bagian karena kemampuan GPU dan RAM yang tersedia tidak mendukung proses training secara penuh. Saat training digunakan memory yang ideal sebesar 25 GB dan menggunakan Tesla P100-PCIE sebagai GPU. Saat testing, index 81 dari dataset utama sebelum dilakukan pembagian split set akan digunakan. Pemilihan index tersebut karena terdapat perbedaan antara model generator dan bukan generator pada jumlah datasetnya. Array 81 adalah index yang terdapat pada testing untuk semua model karena menggunakan random seed yang sama saat pembagian set, tetapi pemilihan index data training kedua jenis model berbeda. Karena gambar berupa 3 dimensi dan nodule selalu berada di posisi tengah, maka ditampilkan gambar 2 dimensi

untuk setiap axis dengan fokus pada sisi tengah axis (jumlah pixel axis terfokus / 2).

Akan ada 4 bagian tersedia pada plot prediksi. Bagian x_test adalah data yang tidak diikutkan dalam proses training. Bagian ground truth adalah bagian nodule yang sudah ditentukan oleh radiologist, bagian predicted adalah hasil prediksi yang ditebak oleh model training dan bagian blended adalah ground truth yang overlapped dengan bagain predicted. Semua perhitungan akurasi testing akan menggunakan jaccard index. Model pertama akan menggunakan arsitektur U-Net biasa dan model kedua menggunakan U-Net++

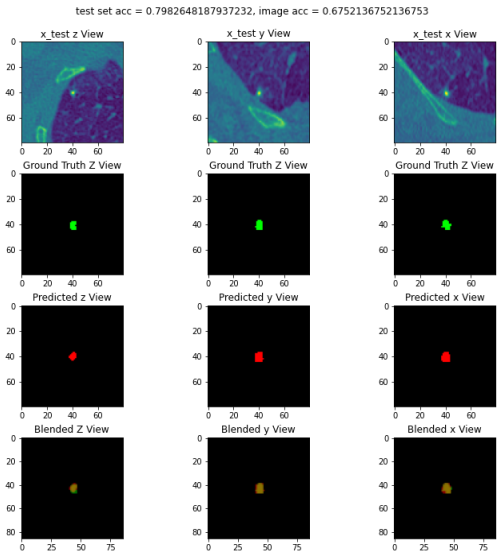
Uji coba hanya akan menggunakan Adam[10] sebagai optimizer saat training. Pemilihan optimizer ini dikarenakan momentum yang dimiliki Adam bersifat dinamis sehingga terdapat lebih sedikit hyperparameter yang perlu disesuaikan dibandingkan optimizer lainnya. Momentum ini akan dipakai untuk mempengaruhi learning rate saat training. Adam bersifat stochastic, yang mana Adam akan memilih salah satu data dari data training untuk dihitung gradientnya lalu melakukan update parameter.

A. Model U-Net



Gambar 18. Hasil Training Model F

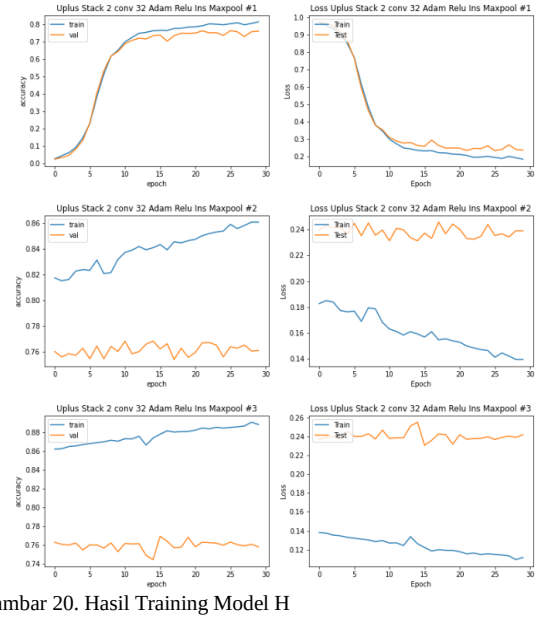
Pada Gambar 18, sesi training dibagi menjadi 4 bagian yang mana terdapat 50 epoch untuk setiap bagian. Proses training dilakukan tanpa menggunakan generator data augmentasi. Dataset yang digunakan adalah 860 gambar dan 400 gambar hasil augmentasi, dengan pembagian 70% untuk training, 20% untuk validation, 10% untuk testing. Saat selesai menjalani 200 epoch, akurasi training berada dikisaran ±91% dan akurasi validasi berada dikisaran ±77%. Pada saat dilakukan prediksi, menggunakan model bagian pertama (50 epoch pertama) menghasilkan akurasi testing sebesar 77.0592 %, bagian kedua menghasilkan akurasi testing sebesar 77.5319%, bagian ketiga menghasilkan akurasi testing sebesar 79.0632%, bagian keempat menghasilkan akurasi testing sebesar 79.8264%. Pada model ini, dilakukan testing menggunakan parameter callback nilai maximum training saat training dan tidak menggunakan parameter, akurasi testing model tanpa callback adalah 79.3363% dan saat menggunakan callback adalah 79.8264. Berdasarkan hasil testing, maka akan dipakai model keempat karena memiliki akurasi terbesar untuk melakukan plotting.



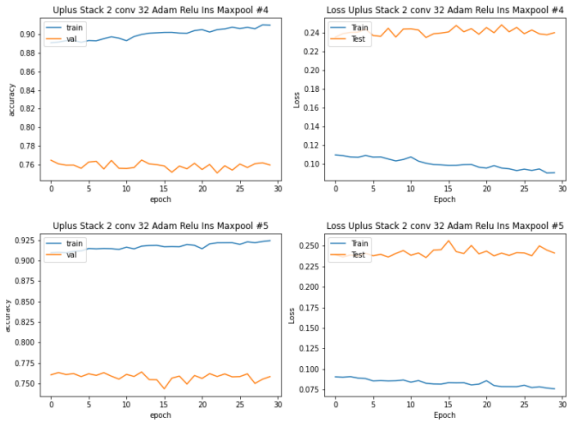
Gambar 19. Contoh Hasil Prediksi Model F

Pada gambar 19, gambar yang digunakan adalah array ke 81 dari dataset utama. Hasil predksi menggunakan Jaccard Index menghasilkan akurasi sebesar 67.5213%. Warna hijau adalah ground truth (segmentasi yang diberikan oleh radiologist), warna merah adalah hasil prediksi, warna coklat adalah bagian intersect antara ground truth dan hasil prediksi. Area ground truth sangat penting karena menentukan tingkat akurasi dari gambar.

B. Model U-Net++

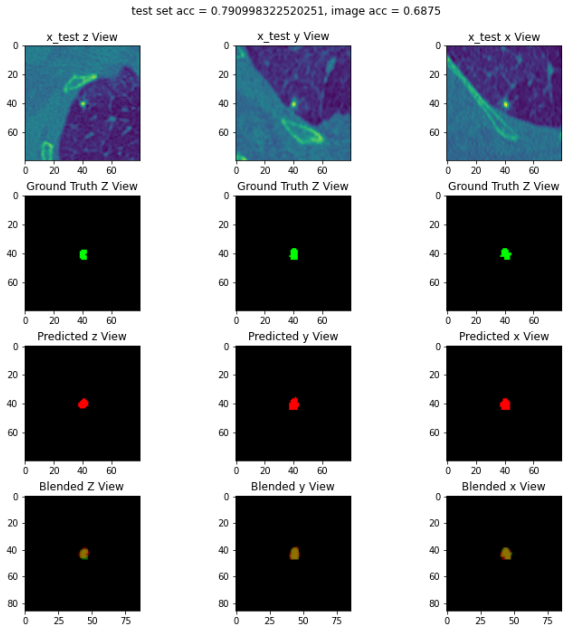


Gambar 20. Hasil Training Model H



Gambar 20. Hasil Training Model H (Lanjutan)

Pada Gambar 20, sesi training dibagi menjadi 5 bagian yang mana terdapat 30 epoch untuk setiap bagian. Proses training dilakukan tanpa menggunakan generator data augmentasi. Dataset yang digunakan adalah 860 gambar dan 400 gambar hasil augmentasi, dengan pembagian 70% untuk training, 20% untuk validation, 10% untuk testing. Saat selesai menjalani 150 epoch, akurasi training berada dikisaran ±92% dan akurasi validasi berada dikisaran ±76%. Pada saat dilakukan prediksi, menggunakan model bagian pertama (50 epoch pertama) menghasilkan akurasi testing sebesar 75.8763%, bagian kedua menghasilkan akurasi testing sebesar 76.3644%, bagian ketiga menghasilkan akurasi testing sebesar 78.4958%, bagian keempat menghasilkan akurasi testing sebesar 79.0998%, bagian kelimat menghasilkan akurasi testing sebesar 78.1955%. Berdasarkan hasil testing, maka akan dipakai model keempat karena memiliki akurasi terbesar untuk melakukan plotting.



Gambar 21. Contoh Hasil Prediksi Model H (Lanjutan)

Pada Gambar 21, gambar yang digunakan adalah array ke 81 dari dataset utama. Hasil prediksi menggunakan Jaccard Index menghasilkan akurasi sebesar 68.75%. Warna hijau adalah ground truth, warna merah adalah hasil prediksi, warna coklat adalah bagian intersect antara ground truth dan hasil prediksi. Kebenaran area ground truth yang diberikan radiologist sangat penting karena menentukan tingkat akurasi dari gambar.

V. KESIMPULAN

Pada subbab ini dijelaskan mengenai kesimpulan dari pembuatan 2 model untuk program prediksi nodule. Nilai skor jaccard index untuk testing model U-Net++ tidak terlalu berbeda jauh hasilnya dari U-Net biasa. Meskipun U-Net++ memiliki komponen layer dan aktivasi yang sama dan memiliki banyak layer daripada U-Net biasa tetap tidak memiliki pengaruh yang signifikan pada akurasi testing.

DAFTAR PUSTAKA

[1] K. Mori, “CAD in lung,” in *Handbook of Medical Image Computing and Computer Assisted Intervention*, Elsevier, 2020, pp. 91–107.

[2] F. Isensee and K. H. Maier-Hein, “An attempt at beating the 3D U-Net,” *arXiv Prepr. arXiv1908.02182*, 2019.

[3] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, “Unet++: A nested u-net architecture for medical image segmentation,” in *Deep learning in medical image analysis and multimodal learning for clinical decision support*, Springer, 2018, pp. 3–11.

[4] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[5] Y. Wu and K. He, “Group normalization,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[6] P. Baldi, “Autoencoders, unsupervised learning, and deep architectures,” in *Proceedings of ICML workshop on unsupervised and transfer learning*, 2012, pp. 37–49.

[7] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, 2015, pp. 234–241.

[8] J. Pedrosa *et al.*, “LNDb: a lung nodule database on computed tomography,” *arXiv Prepr. arXiv1911.08434*, 2019.

[9] G. L. Kindlmann, “An self-contained explanation of image orientation and the ‘measurement frame’, with connections to the NRRD format (Version 0.5),” 2010.

[10] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv Prepr. arXiv1412.6980*, 2014.