SEGMENTASI CITRA BANGUNAN UNTUK MENENTUKAN TINGKAT KERUSAKAN PASCA BENCANA ALAM MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

Oleh:
REVALDI RAHMATMULYA
NIM. 200605110019

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2024

SEGMENTASI CITRA BANGUNAN UNTUK MENENTUKAN TINGKAT KERUSAKAN PASCA BENCANA ALAM MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh:
REVALDI RAHMATMULYA
NIM. 200605110019

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

HALAMAN PERSETUJUAN

SEGMENTASI CITRA BANGUNAN UNTUK MENENTUKAN TINGKAT KERUSAKAN PASCA BENCANA ALAM MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

Oleh:
REVALDI RAHMATMULYA
NIM. 200605110019

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 17 Mei 2024

Pembimbing I,	Pembimbing II,
Dr. Ir. Mokhamad Amin Hariyadi, M.T	Dr. Zainal Abidin, M, M.Kom
NIP. 19670118 200501 1 001	NIP. 19760613 200501 1 004

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT, IPM
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

SEGMENTASI CITRA BANGUNAN UNTUK MENENTUKAN TINGKAT KERUSAKAN PASCA BENCANA ALAM MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

SKRIPSI

Oleh:
REVALDI RAHMATMULYA
NIM. 200605110019

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 20 Juni 2024

Susunan Dewan Penguji

Ketua Penguji	: Dr. Ririen Kusumawati, S.Si., M.Kom NIP. 19720309 200501 2 002	( )
Anggota Penguji I	: Puspa Miladin N S A Basid, M.Kom NIP. 19930828 201903 2 018	( )
Anggota Penguji II	: Dr. Ir. Mokhamad Amin Hariyadi, M.T NIP. 19670018 200501 1 001	( )
Anggota Penguji III	: Dr. Zainal Abidin, M.Kom NIP. 19760613 200501 1 004	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Fachrul Kurniawan, M.MT, IPM
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Revaldi Rahmatmulya
NIM	: 200605110019
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Segmentasi Citra Bangunan untuk Menentukan Tingkat Kerusakan Pasca Bencana Alam Menggunakan *Convolutional Neural Network*

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 17 Mei 2024
Yang membuat pernyataan,

Revaldi Rahmatmulya
NIM. 200605110019

MOTTO

“Apa yang kau tanam, itulah yang akan kau panen”

“Berbuatlah untuk duniamu seakan kamu akan hidup selamanya, dan berbuatlah untuk akhiratmu seakan kamu akan mati besok”

“Hidup adalah pilihan, arah dari kehidupanmu akan ditentukan oleh pilihanmu”

HALAMAN PERSEMBAHAN

Dengan mengucap alhamdulillahi rabbil alamin, saya persembahkan skripsi ini untuk :

1. Kedua orang tua saya Wisnu Nurhariadi, S.T. dan Wiwik Hernamala yang selalu mendoakan saya dan mendukung saya hingga bisa mencapai di titik ini.
2. Guru – guru saya Gus Edy Nuswantoro, Ustad Heri Wahyu Purnomo, Habib Sholeh, Gus Ulin, Kyai Mukhtar, Kyai Ikhsan, karena telah memberikan ilmu yang manfaat dunia akhirat serta senantiasa mendoakan saya.
3. Almarhum guru – guru saya (Alm) Kyai Hamid, (Alm) Habib Abdullah bil faqih, (Alm) Habib Alwi, (Alm) Habib Salim, (Alm) Habib Abdurrohman, (Alm) Kyai Kholil, (Alm) Kyai Abdul faqih, (Alm) Kyai As'ad, (Alm) Kyai Said, (Alm) Kyai Su'aidi, (Alm) Asrori, (Alm) Kyai Zainullah, (Alm) Kyai Jamal, (Alm) Kyai Muttaqin, (Alm) Kyai Ismail, (Alm) Kyai Munawwir, (Alm) Kyai Arwani, (Alm) Kyai Muntaha, (Alm) Ustad Musa, (Alm) Kyai Sulaiman, (Alm) Kyai Hanan, (Alm) Kyai Abu Amar, (Alm) Kyai Ahmad Shiddiq, (Alm) Kyai Musta'in, (Alm) Mbah Kyai Khusnan, (Alm) Mbah Kyai Thoyyib, (Alm) Mbah Kyai Jazuli, karena berkat barokah sanad keilmuan beliau dan rahmat Allah SWT saya dapat sampai di titik ini.
4. Kedua adik tersayang saya Kyla Anadilareva dan Ahza Revandila.
5. Teman – teman saya khususnya "INTEGER Teknik Informatika Angkatan 20" karena selalu mendukung, membantu, dan menemani saya baik dalam keadaan susah maupun senang.
6. Teman – teman dari komunitas Weboender Community yang sudah memberikan ilmu khususnya pada pemrograman website dan juga selalu mendukung saya.

KATA PENGANTAR

Assalamualaikum Warahmatullahi Wabarakatuh.

Segala puji hanya milik Allah Subhanahu Wa Ta'ala atas segala nikmat dan kasih sayang-Nya yang telah memudahkan penulis untuk menyelesaikan skripsi yang berjudul "Segmentasi Citra Bangunan untuk Menentukan Tingkat Kerusakan Pasca Bencana Alam Menggunakan Convolutional Neural Network". Semoga shalawat dan salam senantiasa terlimpah kepada Nabi Muhammad Sallalahu 'Alaihi wa Sallam. Dan semoga kita semua mendapat syafaatnya di hari kiamat nanti, Aamiin.

Penulis mengucapkan rasa syukur dan terima kasih yang tak terhingga kepada semua pihak-pihak yang selalu memberikan bantuan dan motivasi kepada penulis untuk dapat menyelesaikan skripsi ini. Ucapan ini penulis sampaikan kepada:

1. Prof. Dr. H. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Hariani, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Fachrul Kurniawan, M.MT., IPM, selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Dr. Ir. Mokhamad Amin Hariyadi, M.T., selaku dosen pembimbing I dan Dr. Zainal Abidin, M, M.Kom selaku dosen pembimbing II yang telah memberikan bantuan dan arahan kepada penulis, sehingga bisa menuntaskan skripsi ini.

5. Dr Ririen Kusumawati, S.Si., M.Kom selaku dosen penguji I dan Puspa Miladin Nuraida Safitri A Basid, M.Kom selaku dosen penguji II yang telah menguji serta memberikan masukan sehingga penulis dapat menuntaskan skripsi dengan baik.
6. Segenap Dosen, Admin, Laboran dan Mahasiswa Program Studi Teknik Informatika yang telah memberikan banyak dukungan dan bimbingan selama pengerjaan skripsi ini.
7. Papa, Mama, serta adik - adik saya yang selalu memberikan dukungan dan motivasi untuk terus berusaha, dan doa yang tak putus-putusnya selalu disampaikan agar dapat menuntaskan skripsi ini dengan lancar dan baik.

Akhir kata, penulis mengakui bahwa penulisan pada skripsi ini masih banyak kekurangan. Saya berharap semoga skripsi ini diterima sebagai amal ibadah yang tulus dan bermanfaat di sisi Allah Subhanahu Wa Ta'ala. Semoga karya ini menjadi bagian dari kontribusi yang tak terputus dalam rangka memperkuat dan mengembangkan ilmu pengetahuan, serta melaksanakan tugas sebagai hamba Allah yang berkomitmen.

Wassalamualaikum Warahmatullahi Wabarakatuh.

Malang, 17 Mei 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

ABSTRAK

Rahmatmulya, Revaldi. 2024. **Segmentasi Citra Bangunan untuk Menentukan Tingkat Kerusakan Pasca Bencana Alam Menggunakan *Convolutional Neural Network***. Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. Ir. Mokhamad Amin Hariyadi, M.T. (II) Dr. Zainal Abidin, M.Kom

Kata kunci: *Convolutional Neural Network, Machine Learning, Segmentasi.*

Bencana alam merupakan suatu kejadian yang disebabkan oleh alam seperti halnya gempa bumi, angin tornado, tsunami, kebakaran hutan, dan lain - lain. Dampak yang diberikan karena terjadinya bencana alam sangat besar dan beragam di berbagai sektor baik itu sektor ekonomi, kesehatan, dan utamanya adalah bagunan. Dibutuhkan Tindakan yang efektif dan efisien guna membantu pemulihan pasca bencana alam, salah satunya dengan membantu identifikasi tingkat kerusakan bangunan pasca bencana alam. Untuk mengatasi permasalahan ini, pada penelitian ini akan dirancang system yang mampu melakukan segmentasi guna mengetahui tingkat kerusakan suatu bangunan pasca bencana alam menggunakan metode convolutional neural network .Data yang digunakan merupakan data aerial image yang bersumber dari xView2: Assess Building Damage yang berisikan 50 aerial image dengan terdapat 5 kelas yaitu no-damage, minor-damage, major-damage, destroyed dan unlabeled. Langkah yang dilakukan pada penelitian ini meliputi preprocessing data dengan menggunakan patchify dan augmentasi data. Kemudian lenjut ekstrasi fitur dengan konvolusi dan dilanjutkan dengan proses training menggunakan neural network menggunakan arsitektur yang diajukan. Penelitian ini mengajukan arsitektur dengan 27 hidden layer dengan fitur ekstrasi menggunakan *average pooling*. Proses evaluasi model akan menggunakan Mean Intersection over Union (MIoU) untuk melihat seberapa mirip bentuk hasil prediksi segmentasi dengan data aslinya. Hasil pengujian yang telah dilakukan dengan total 8 skenario dengan 4 skenario *average pooling* dan 4 skenario max pooling, arsitektur yang diajukan mampu memberikan hasil MIoU terbaik dengan nilai 0.31 dan akurasi sebesar 0.9577.

ABSTRACT

Rahmatmulya, Revaldi. 2024. **Building Image Segmentation for Determining Post-Disaster Damage Level Using Convolutional Neural Network.** Undergraduate Thesis. Informatics Engineering Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisors: (I) Dr. Ir. Mokhamad Amin Hariyadi, M.T. (II) Dr. Zainal Abidin, M.Kom

Natural disasters are events caused by nature such as earthquakes, tornadoes, tsunamis, forest fires, and others. The impacts of natural disasters are significant and varied across various sectors, including the economy, health, and primarily, infrastructure. Effective and efficient actions are needed to assist in the recovery following natural disasters, one of which is aiding in the identification of building damage levels post-disaster. To address this issue, this research proposes a system capable of performing segmentation to determine the level of building damage post-natural disaster using convolutional neural network methods. The data utilized consists of aerial images sourced from xView2: Assess Building Damage, comprising 50 aerial images with 5 classes: no-damage, minor-damage, major-damage, destroyed, and unlabeled. The steps undertaken in this research include data preprocessing using patchify and data augmentation. Subsequently, feature extraction is performed using convolution, followed by the training process using a neural network with the proposed architecture. This study proposes an architecture with 27 hidden layers, with feature extraction utilizing *average pooling*. The model evaluation process will employ Mean Intersection over Union (MIoU) to assess how closely the segmentation prediction results resemble the original data. The testing results conducted encompass a total of 8 scenarios, with 4 scenarios utilizing *average pooling* and 4 scenarios utilizing max pooling. The proposed architecture demonstrates the best MIoU result with a value of 0.31 and an accuracy of 0.9577.

Keywords: *Convolutional Neural Network, Machine Learning, Segmentation*

مستخلص البحث

رحمة موليا، ريفالدي. 2024. تقسيم صور المباني لتحديد مستوى الضرر بعد الكوارث الطبيعية باستخدام الشبكات العصبية التصاعدية. رسالة بكالوريوس. برنامج الهندسة الحاسوبية، كلية العلوم والتكنولوجيا، جامعة الدولة الإسلامية مولانا مالك إبراهيم مالانغ. المشرفون: (١) الدكتور محمد أمين حاريادي، م.ت. (٢) الدكتور زين العابدين، م.كوم.

الكلمات الرئيسية: التقسيم، الشبكة العصبية التصاعدية، التعلم الآلي

الكوارث الطبيعية هي حوادث تسببها الطبيعة مثل الزلازل، الأعاصير، التسونامي، حرائق الغابات، وغيرها. تترتب عن وقوع الكوارث الطبيعية آثار كبيرة ومتنوعة في مختلف القطاعات، سواء كانت الاقتصادية أو الصحية، والأهم من ذلك هو القطاع العقاري. يتطلب التعامل بفعالية وكفاءة لمساعدة في إعادة بناء ما بعد الكوارث الطبيعية، ومن ضمن الأساليب التي يمكن أن تساعد في ذلك هو تحديد مستوى الضرر في البنية التحتية بعد الكوارث الطبيعية. لحل هذه المشكلة، سيتم تصميم نظام قادر على تقسيم الصور لتحديد مستوى الضرر في البنية التحتية بعد الكوارث الطبيعية باستخدام طريقة الشبكة العصبية التصاعدية. يستخدم البيانات التي تم الحصول عليها صور جوية من مصدرxView2: Assess Building Damage ، والتي تحتوي على 50 صورة جوية مصنفة إلى 5 فئات وهي: لا يوجد ضرر، ضرر طفيف، ضرر كبير، تدمير وغير مصنف. تشمل خطوات البحث هذه تجهيز البيانات باستخدام patchify وزيادة البيانات، ثم استخراج الميزات بواسطة التحصيص ومن ثم عملية التدريب باستخدام الشبكة العصبية باستخدام الهندسة المعمارية المقترحة. يقدم البحث هذا هندسة مع 27 طبقة مخفية مع استخراج الميزات باستخدام التجميع المتوسط. سيتم استخدام معيار انترسكشن أوفر يونيون (MIoU) لتقييم مدى تشابه نتائج التحديد مع البيانات الأصلية. أظهرت نتائج الاختبارات التي تمت بمجموع 8 سيناريوهات، 4 منها بتجميع متوسط و 4 بتجميع أقصى، أن الهندسة المعمارية المقترحة قادرة على تحقيق أفضل قيمة MIoU بقيمة 0.31 ودقة .0.9577

BAB I

PENDAHULUAN

1.1 Latar Belakang

Bencana alam adalah peristiwa yang disebabkan oleh alam seperti gempa bumi, tsunami, banjir, kebakaran hutan, badai, kekeringan, dan gelombang panas. Peristiwa - peristiwa ini menjadi salah satu bagian dari *riskcape* yang mana manusia telah mempelajarinya untuk bisa hidup berdampingan dengannya. Akan tetapi, dampak dari bencana alam terus mengalami peningkatan yang signifikan dalam beberapa tahun terakhir (Rosselló, Becken, and Santana-Gallego, 2020). Dampak yang diberikan karena terjadinya bencana alam sangat besar dan beragam di berbagai sektor baik itu sektor ekonomi, kesehatan, bangunan, dan sektor vegetasi (Padli, Habibullah, and Baharom, 2010). Karena banyaknya sektor yang terdampak akibat bencana alam, dibutuhkan penanganan yang efisien dan efektif.

Dalam hal ini, salah satu sektor yang penting untuk ditangani adalah sektor fisik terutama dalam penanganan bangunan yang rusak pasca bencana alam. Ketika sebuah bangunan rusak akibat bencana alam, penting untuk menilai sejauh mana kerusakan yang diakibatkan oleh bencana alam tersebut. Hal ini guna menentukan sejauh mana perbaikan yang diperlukan. Tingkat kerusakan dapat bervariasi tergantung dari jenis bencana alam dan lokasi dari bangunan tersebut. Dalam beberapa kasus, kerusakan ringan hanya akan memerlukan sedikit perbaikan dan biaya yang tidak besar, namun kerusakan juga bisa parah dan memerlukan biaya yang tinggi untuk melakukan perbaikan bahkan rekonstruksi ulang (Pramono *et al*., 2022).

Pengukuran tingkat kerusakan bangunan pasca bencana alam secara seringkali dilakukan dengan cara manual, hal ini tentu tidak efektif dalam penentuannya. Terdapat beberapa faktor yang menjadi penyebab mengapa penentuan secara manual terbilang tidak efektif, yang pertama memerlukan banyak tenaga dan waktu. Butuhnya banyak orang dikarenakan kerusakan yang terjadi berdampak tidak hanya satu bangunan, hal ini tentu saja membuat waktu penentuan menjadi semakin panjang. Akibatnya bisa menghambat respon dari tim penyelamat (Zaryabi *et al*., 2022). Berikutnya yang kedua yaitu kebergantungan kepada ahli dalam penentuan tingkat kerusakan bangunan pasca bencana alam juga menjadi salah satu faktor. Dibutuhkan sorang ahli agar penentuan tingkat kerusakan menjadi konsisten, namun keterbatasan sang ahli menjadi penghambat dalam penentuannya (Pahlavani, Samadzadegan, and Delavar, 2006).

Untuk mengatasi keterbatasan ini, model pengembangan *model deep learning* dibutuhkan untuk secara otomatis bisa menentukan tingkat kerusakan bangunan pasca bencana alam (Zaryabi *et al*., 2022). Penulis menggunakan model *deep learning* dengan metode *covolutional neural network* menggunakan arsitektur yang telah penulis rancang dalam pendekatan komputasi untuk menentukan tingkat kerusakan bangunan pasca bencana alam. Dengan memanfaatkan data – data berupa citra bangunan yang rusak akibat bencana alam, model *deep learning* ini memliki potensi untuk menentukan tingkat kerusakan pasca bencana alam yang efektif dan efisien. Dengan demikian, diharapkan dampak – dampak yang terjadi akibat dilakukannya cara manual dalam penentuan tingkat kerusakan bangunan pasca bencana alam tidak sampai terjadi.

Penggunaan model *deep learning* untuk penentuan tingkat kerusakan bangunan pasca bencana alam memiliki potensi yang besar untuk memberikan jalan dalam pengembangan model segmentasi yang lebih kompleks dan selaras dengan konsep islam, bahwa penggunaan model *deep learning* berarti sama dengan melakukan pekerjaan dengan akurat.

Ayat al-Qur'an yang dapat dijadikan acuan dalam bekerja secara akurat dan tepat terdapat pada Surat al-Isra ayat 36 yang berbunyi :

وَلَا تَقْفُ مَا لَيْسَ لَكَ بِهِۦ عِلْمٌ ۚ إِنَّ ٱلسَّمْعَ وَٱلْبَصَرَ وَٱلْفُؤَادَ كُلُّ أُو۟لَٰٓئِكَ كَانَ عَنْهُ مَسْـُٔولًا

"*Dan janganlah kamu mengikuti apa yang kamu tidak mempunyai pengetahuan tentangnya. Sesungguhnya pendengaran, penglihatan dan hati, semuanya itu akan diminta pertanggungan jawabnya." (QS. Al – Isra: 36)*

Tafsir ayat ini yang berasal dari Kementrian Agama RI berbunyi "Dan janganlah kamu mengikuti apa yang kamu tidak mempunyai pengetahuan tentangnya. Jangan mengatakan sesuatu yang engkau tidak ketahui, jangan mengaku melihat apa yang tidak engkau lihat, jangan pula mengaku mendengar apa yang tidak engkau dengar, atau mengalami apa yang tidak engkau alami. Sesungguhnya pendengaran, penglihatan dan hati, adalah amanah dari tuhanmu, semuanya itu akan diminta pertanggunganjawabnya, apakah pemiliknya menggunakan untuk kebaikan atau keburukan dan janganlah engkau berjalan di muka bumi ini dengan sombong, untuk menampakkan kekuasaan dan kekuatanmu, karena sesungguhnya sekuat apa pun hentakan kakimu, kamu sekali-kali tidak dapat menembus bumi dan setinggi apa pun kepalamu, sekali-kali kamu tidak akan sampai setinggi gunung. Sesungguhnya kamu adalah makhluk yang

lemah dan rendah di hadapan Allah, kamu tidak memiliki kekuatan dan kemuliaan, melainkan apa yang dianugerahkan oleh-Nya." (Kemenag RI, 2016a)

Tafsir QS. Al-Isra ayat 36 mengajarkan pentingnya berpegang pada pengetahuan yang benar dan akurat sebelum memberikan penilaian atau menilai sesuatu. Ayat ini juga mengajarkan pentingnya tidak sombong dan merendahkan diri di hadapan Allah. Secara umum ayat ini mengajarkan bahwa manusia harus berpegang pada pengetahuan yang benar dan akurat sebelum mengambil tindakan atau memberikan penilaian. Dalam konteks penelitian, ayat ini dapat diintegrasikan dengan menggunakan metode *convolutional neural network* yang akurat dan teliti dalam menentukan tingkat kerusakan bangunan pasca bencana alam. Dengan demikian, penelitian dapat menghasilkan keputusan yang akurat dan dapat dipertanggungjawabkan.

Penelitian mengenai penentuan tingkat kerusakan bangunan pasca bencana alam pernah dilakukan dengan menggunakan metode *artificial nerual network* (Almais *et al*., 2023). Dalam penelitian ini, peneliti menggunakan data teks dan tidak menggunakan data citra. Hasil percobaan menunjukkan bahwa model pola data E5 memiliki tingkat akurasi ideal sebesar 97 persen dengan nilai *Mean Squared Error (MSE)* sebesar 0.06 dan *Mean Absolute Percentage Error (MAPE)* sebesar 3 persen. Berdasarkan penelitian sebelumnya penulis menganggap masalah ini bisa dioptimalkan dengan data citra menggunakan metode *convolutional neural network*.

Berikutnya penelitian mengenai segmentasi kerusakan menggunakan metode *covolutional neural network* dengan arsitektur Unet pernah dilakukan

dalam permasalahan penyakit pada tanaman padi yang terkena hama leafblast (Annafii *et al*., 2022). Dalam penelitian ini, berdasarkan total 300 data yang terdampak penyakit dapat di segmentasi dengan baik dengan menghasilkan akurasi model sebesar 98,60 dengan loss hanya 0,0526. Hal ini menunjukkan bahwa penggunaan model *U-Net* dalam proses segmentasi terutama dalam hal ini adalah segmentasi kerusakan memberikan hasil yang sangat baik.

Dengan dasar masalah yang telah diuraikan, alasan pemilihan metode *Convolutional Neural Network* didasarkan pada penelitian terdahulu. Studi Almais *et al*. (2023) menunjukkan keberhasilan artificial neural network dalam menentukan tingkat kerusakan bangunan pasca bencana, namun dengan data teks. Ini menginspirasi penggunaan data citra untuk optimalisasi analisis, dengan *Convolutional Neural Network* dikenal efektif dalam data visual. Penelitian Annafii *et al*. (2022) mendukung pemilihan *convolutional neural network*, terutama arsitektur U-Net, dalam segmentasi kerusakan pada tanaman padi. Keberhasilan segmentasi tersebut menegaskan potensi *convolutional neural network* dalam pemisahan objek dari citra dengan akurasi tinggi. Melalui penelitian ini, penulis bertujuan menerapkan *Convolutional Neural Network* dalam segmentasi bangunan untuk menentukan tingkat kerusakan pasca bencana alam.

1.2 Identifikasi Masalah

Bagaimana mengukur nilai akurasi dan MIoU yang dihasilkan dari metode *convolutional neural network* dalam segmentasi citra bangunan rusak pasca bencana alam untuk menentukan tingkat kerusakan bangunan ?

1.3 Batasan Masalah

1. Penelitian ini mengukur tingkat kerusakan bangunan, bukan korban jiwa.
2. Menggunakan model *convolutional neural network* untuk segementasi citra.

1.4 Tujuan Penelitian

Mendapatakan hasil segmentasi kerusakan bangunan untuk menilai tingkat kerusakannya.

1.5 Manfaat Penelitian

1. Mempermudah kinerja tim penanganan pasca bencana alam dalam menentukan tingkat kerusakan bangunan, sehingga dapat mengoptimasi pengambilan keputusan.
2. Potensi penelitian lebih mendalam mengenai segmentasi kerusakan bangunan dengan metode *convolutional neural network*.

BAB II

STUDI PUSTAKA

2.1 Bencana Alam

Bencana alam adalah peristiwa alam yang tidak terduga dan dapat menimbulkan kerusakan pada lingkungan dan kehidupan manusia. Bencana alam dapat terjadi karena berbagai faktor, seperti perubahan iklim, aktivitas vulkanik, gempa bumi, banjir, dan tanah longsor. Bencana alam dapat menyebabkan kerusakan pada infrastruktur, bangunan, dan fasilitas publik, serta menimbulkan korban jiwa dan kerugian ekonomi yang besar (Nascimento *et al*., 2017).

Salah satu dampak bencana alam yaitu kerusakan bangunan pasca bencana alam dapat terjadi pada berbagai jenis bencana alam, seperti gempa bumi, banjir, tanah longsor, dan tsunami. Kerusakan pada bangunan dapat terjadi pada bagian struktural, seperti pondasi, dinding, dan lantai, serta pada bagian non-struktural, seperti pintu, jendela, dan perabotan. Tingkat keruskaan bangunan juga bervariasi tergantung seberapa parah kerusakan yang diakibatkan oleh bencana alam terhadap bangunan tersebut.

Kerusakan bangunan pasca bencana alam dapat berdampak signifikan pada kehidupan manusia di daerah terdampak. Evaluasi tingkat kerusakan struktur bangunan menjadi penting untuk memahami kondisi pasca bencana. Penelitian Rehan, Bastian, and Kurniawan (2023) menyoroti urgensi analisis tersebut dalam konteks penelitian kondisi bangunan pasca bencana. Dalam hal ini, penentuan tingkat kerusakan menjadi fokus penting, karena dapat memberikan informasi yang berharga terkait kesiapan daerah terhadap perbaikan dan rehabilitasi pasca

bencana. Oleh karena itu, pemilihan metode yang tepat untuk analisis tingkat kerusakan bangunan sangat diperlukan, dan inilah mengapa penerapan *Convolutional Neural Network* menjadi pilihan yang strategis dalam penelitian ini.

2.2 Convolutional Neural Network

Convolutional Neural Network (CNN) adalah salah satu jenis jaringan saraf tiruan yang digunakan untuk memproses data berupa citra atau video. CNN menggunakan matriks kernel konvolusional untuk mengekstrak fitur dari data masukan dan menghasilkan keluaran yang diinginkan. *convolutional neural network* terdiri dari beberapa layer, seperti *convolution layer*, *pooling layer*, dan *fully connected layer*, yang masing-masing lapisan mempunyai fungsi khusus dalam mengolah data masukan (LeCun, Bengio, and Hinton, 2015).

CNN menunjukkan performa yang sangat bagus dalam kontribusi khususnya pada bidang *computer vision* dan *image processing*. Hal ini dibuktikan dengan penggunaan CNN yang sangat populer berkat kemampuannya yang mampu digunakan dalam banyak hal pada *machine learning*. Area cakupan CNN diantaranya mengenai *image segmentation*, *image classification*, *object detection*, *video processing*, *natural language processing*, dan *speech recognition* (Khan *et al*., 2019).

2.3 Segmentasi *Convolutional Neural Network*

Pada penelitian yang dilakukan oleh Sreekumar and Geetha (2020) yang meneliti mengenai segmentasi citra tangan manusia. Peneliti mengangkat masalah

ini karena berkeyakinan untuk bisa dikembangkan lebih dalam lagi khususnya dalam hal *motion recognition*. Peneliti memilih menggunakan metode *convolutional neural network* dengan arsitektur *U-Net* dalam mensegmentasi citra tangan manusia. Hasil dari penelitian ini, model yang diciptakan menggunakan arsitektur *U-Net* mampu memperoleh akurasi sebesar 0.98 sedangkan arsitektur lain yang digunakan sebagai pembanding yaitu fcn8, fcn8_vgg, segnet, segnet_vgg memliki akurasi secara berturut – turut 0.95, 0.91, 0.85, 0.89. Dengan ini peneliti mempunyai kesimpulan bahwa arsitektur *U-Net* mampu mensegmentasi citra tangan manusia bahkan dengan latar belakang citra yang kompleks.

Selanjutnya penelitian yang dilakukan oleh Karki and Kulkarni (2021) membahas mengenai segmentasi kapal laut. Masalah yang diangkat peneliti yaitu akibat dari tindakan kriminal dan terlarang yang tidak hanya terjadi di daratan namun juga dilautan, oleh karena itu menurut penulis dibutuhkan otomasi untuk bisa mendeteksi adanya kapal laut. Model terbaik yang berhasil dibuat dalam eksperimen yang dilakukan peneliti adalah model dari arsitektur *U-Net* dengan *f2 score* sebesar 0.823.

Berikutnya adalah penelitian yang dilakukan oleh Sivagami *et al* (2020) yang meneliti tentang segmentasi citra gigi. Dengan penelitian ini, peneliti berharap dokter gigi akan terbantu untuk bisa mendeteksi mana gigi yang rusak, menentukan tempat yang tempat untuk impan gigi, dan menentukan struktur gigi yang tepat. Data yang digunakan oleh peneliti adalah citra gigi *radiographic* yang didapatkan melalui *X-Ray, Computed Tomography (CT) dan Magnetic resonance*

imaging (MRI). Peneliti memlih untuk menggunakan metode segmentasi *U-Net* dalam penelitiannya, tetapi ia juga menyertakan hasil dari segmentasi metode lain sebagai pembanding yaitu *Global thresholding, Fuzzy C-Means, Watershed dan Canny edge detection.* Hasil dari penelitian ini yaitu *U-Net* menempati posisi pertama sebagai metode terbaik dengan akurasi sebesar 0.97. Sedangkan akurasi model dari metode segmentasi lainnya secara berturut – turut yaitu 0.79, 0.82, 0.77, 0.79. Dengan ini peneliti memberikan kesimpulan bahwa segmentasi yang dilakukan oleh *U-Net* sangat akurat untuk bisa mensegmentasi citra *radiogrhrapic* gigi.

Berikutnya penelitian yang dilakukan oleh Gangurde (2023) yang meneliti tentang segmentasi bangunan dan jalanan kota dari citra peta yang didapatkan melalui satelit dan UAV. Pada penelitian ini , peneliti mengusulkan untuk menggunakan arsitektur U-Net yang sudah dimodifikasi yaitu dengan mengganti encoder dari U-Net menjadi EfficientNetV2L untuk mensegmentasi bangunan dan EfficientNetB7 untuk mensegmentasi jalanan. Hasil yang diperoleh sangat memusakan karena peneliti berhasil mendapatkan mIoU dan *F1-Score* secara berturut – turut untuk segmentasi bangunan sebesar 0.08365 dan 0.9104. Lalu hasil mIoU dan *F1-Score* untuk segmentasi jalanan secara berturut – turut yaitu 0.9153 dan 0.9556. Dengan ini, peneliti berkesimpulan bahwa U-Net dengan *pre-trained encoder* adalah yang terbaik untuk digunakan dalam segmentasi bangunan dan jalanan.

Penelitian selanjutnya yaitu dilakukan oleh El Rai *et al* (2020) dalam mensegmentasi tumpahan minyak bumi pada citra lautan. Tumpahan minyak

terlihat berwarna hitam pada sensor radar, maka disini peneliti memiliki tantangan untuk bisa memisahkan mana yang tumpahan minyak dan mana yang bukan dengan menggunakan *semantic segmentation pixel level* menggunakan U-Net. Disini peneliti mengusulkan penggunaan U-Net + *Active Contours Without Edge* (ACWE) dalam melakukan segmentasi citra lautan. Hasil yang diperoleh cukup memuaskan dimana metode yang diusulkan peneliti mendapatkan nilai mIoU sebesar 68.09. Ini lebih besar dibandingkan jika hanya menggunakan U-Net saja yang hanya mendapatkan nilai mIoU sebesar 65.49. Peneliti berkesimpulan bahwa U-Net yang mneggunakan ACWE sebagai *loss function* mampu mengkombinasikan panjang dari edge dengan regions yang serupa, yang mana hal ini mampu memberikan peningkatan pada akurasi model.

Penelitian mengenai segmentasi tumor pada otak manusia yang diakukan oleh Aghalari, Aghagolzadeh, and Ezoji (2021) dengan menggunakan U-Net juga membuahkan hasil yang memuaskan. Pada penelitian ini, modifikasi U-Net yang dimodifikasi dengan menambahkan *two-path-way-residual (TPR)* berhasil menghasilkan nilai akurasi sebesar 99.87 . Oleh karena itu peneliti berskesimpulan bahwa model yang diajukan peneliti memberikan keuntugan berupa biaya komputasi yang berkurang, segmentasi yang lebih cepat, dan tidak menggunakan post-processing.

Berikutnya penelitian oleh Annafii *et al* (2022) yang meneliti tentang kerusakan pada tanaman padi. Disini peneliti ingin melakukan segmentasi pada kerusakan padi yang terkena hama *leafblast*. Metode yang diusulkan oleh peneliti adalah U-Net, namun disini peneliti juga ingin melakukan optimasi dengan

Hyperband untuk membuahkan hasil yang lebih maksimal dari U-Net biasa. Hasilnya akurasi dengan menggunakan U-Net menghasilkan sebesar 98.48 sedangkan U-Net dengan optimasi *Hyperband* menghasilkan akurasi sebesar 98.70. Dengan ini peneliti berkesimpulan bahwa segmentasi kerusakan pada padi dengan menggunakan metode *convolutional neural network* model U-Net memberikan hasil yang sangat baik. Lalu optimasi dengan *Hyperband* untuk model U-Net juga telah berhasil mengurangi jumlah parameter menjadi 10.936.760 dan meningkatkan akurasi model menjadi 98.70.

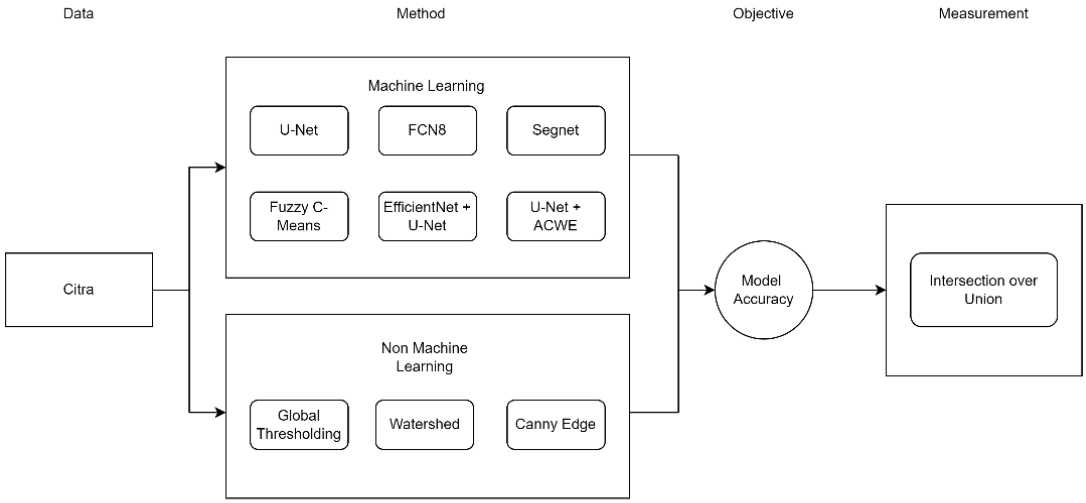
Selanjutnya yaitu penelitian yang berusaha mensegmentasi keretakan pada citra jalanan rusak yang dilakukan oleh Di Benedetto, Fiani, and Gujski (2023). Penelitian ini mengusulkan model modifikasi U-Net dengan mengganti encodernya dengan RestNet50. Hasil yang memuaskan berhasil didapatkan oleh peneliti dengan medapatkan nilai precission, *f1-score* dan mIoU secara berturut 0 turut sebesar 0.8534 , 0.7577, 0.6248. Dengan ini peneliti berkesimpulan bahwa modifikasi U-Net dengan mengganti encodernya menjadi RestNet50 memberikan hasil yang sangat baik untuk segmentasi keretakan pada citra jalanan rusak.

Selanjutnya penelitian mengenai segmentasi bangunan, lahan vegetasi, lahan tandus, dan perairan pada citra peta yang dilakukan oleh Kotaridis and Lazaridou (2022) dengan menggunakan metode *convolutional neural network* model U-Net memberikan hasil yang sangat baik. Peneliti berhasil menghasilkan model dengan akurasi sebesar 0.97 dalam mensegmentasi. Dengan ini peneliti berkesimpulan model U-Net mampu melakukan segmentasi pada citra peta dengan sangat baik.

Terakhir yaitu penelitian yang dilakukan oleh Irwansyah, Heryadi, and Santoso Gunawan (2020) dalam bahasannya tentang segmentasi bangunan atau bukan bangunan pada citra peta. Peneliti mengusulkan untuk menggunakan model *convolutional neural network* dan model U-Net dalam penelitiannya. Hasil yang didapatkan oleh peneliti yaitu model yang mempunyai nilai akurasi sebesar 0.83. Dengan ini peneliti berkesimpulan bahwa segemntasi untuk menentukan bangunan atau bukan dengan menggunakan metode *convolutional neural network* model U-Net dapat dilakukan dengan sangat baik.

2.4 Kerangka Teori

Dalam melakukan penelitian tentang penentuan tingkat kerusakan bangunan pasca bencana alam dengan metode *convolutional neural network* perlu mengacu pada jurnal-jurnal sebelumnya yang telah melakukan sebagai kerangka teori seperti yang terlihat pada Gambar 2.1 :



Gambar 2.1 *Teoritical Framework*

Pada gambar 2.1 dapat dilihat terdapat dua pendekatan utama dalam segmentasi citra yang digunakan dalam jurnal-jurnal sebelumnya yang telah melakukan penelitian terkait, yaitu Machine Learning (ML) dan Non Machine Learning. Data disini semuanya berbentuk citra yang nantinya di dalukan segmentasi dengan metode tertentu. Di bawah pendekatan Machine Learning, seperti U-Net, FCN8, Segnet, Fuzzy C-Means, EfficientNet + U-Net, U-Net + ACWE digunakan untuk menciptakan model yang dapat melakukan segmentasi dengan baik. Di sisi lain, pendekatan Non Machine Learning melibatkan metode Global Thresholding, Watershed dan Canny Edge untuk mensegmentasi citra. Berikutnya pada tahap evaluasi disini menggunakan pengukuran teknik matrik Intersection Over Union yang akan mengukur model. Tabel kerangka teori yang dapat dilihat pada Tabel 2.1 :

Tabel 2.1 Kerangka Teori

No.	Peneliti (Tahun)	Studi Kasus	Metode Penelitian	Hasil Penelitian
1.	Sreekumar and Geetha (2020)	Segmentasi citra tangan manusia dengan latar belakang citra yang kompleks.	- U-Net - fcn8 - fcn8_vgg - segnet - segnet_vgg	0.98 0.95 0.91 0.85 0.89
2.	Karki and Kulkarni (2021)	Segmentasi kapal laut pada citra laut.	CNN arsitektur U-Net	F2-Score 0.823
3.	Sivagami *et al.* (2020)	Segmentasi citra radiographic gigi untuk membantu dokter gigi.	- U-Net - Global thresholding - Fuzzy C-Means - Watershed - Canny edge detection.	0.97 0.79 0.82 0.77 0.79
4.	Gangurde (2023)	Segmentasi bangunan dan jalanan dari satelit dan UAVs.	EfficientNetV2L + UNet (untuk data bangunan) EfficientNetB7 + UNet	0.9122 0.9566

No.	Peneliti (Tahun)	Studi Kasus	Metode Penelitian	Hasil Penelitian
5.	El Rai *et al.* (2020)	Segmentasi tumpahan minyak bumi pada citra lautan.	Unet Unet + ACWE	mIoU : 65.49 mIoU : 68.09
6.	Aghalari *et al.* (2021)	Segmentasi tumor pada citra otak manusia.	TPRE U-Net TPRD U-Net TPRED U-Net	99.87 99.87 99.87
7.	Annafii *et al.* (2022)	Segmentasi kerusakan pada citra padi.	U-Net U-Net dioptimasi dengan Hyperband	0.9848 0.9722
8.	Di Benedetto *et al.* (2023)	Segmentasi retakan pada citra jalanan rusak.	U-Net	IoU : 0.6248
9.	Kotaridis and Lazaridou (2022)	Segmentasi bangunan, lahan vegetasi, lahan tandus, dan perairan pada citra peta.	U-Net	0.97
10.	Irwansyah *et al.* (2020)	Segmentasi penentuan bangunan atau bukan pada citra peta.	U-Net	0.83

Berdasarkan Tabel 2.1, penggunaan metode *Convolutional Neural Network* dalam melakukan segmentasi citra menunjukkan kinerja yang optimal dalam berbagai domain, termasuk bidang medis dan non-medis seperti infrastruktur dan objek. Namun, peneliti belum menemukan penelitian sebelumnya yang secara khusus memanfaatkan metode *convolutional neural network* dalam konteks studi kasus segmentasi kerusakan bangunan pasca bencana alam. Oleh karena itu, penelitian ini mengeksplorasi penerapan *convolutional neural network* sebagai pendekatan eksperimental untuk menganalisis kerusakan bangunan pasca bencana alam.

BAB III

DESAIN DAN IMPLEMENTASI

Pada bab ini desain dan implementasi ini, penulis akan membahas semua hal yang akan berhubungan dengan persiapan peneltian, apa saja yang akan dilakukan, dan bagaimana cara kerja metode yang diusulkan dalam penelitian ini. Bab ini akan berisi rancangan desain sistem yang akan berbicara mengenai alur dari proses segmentasi oleh arsitektur yang diajukan dengan menggunakan metode *convolutional neural network*. Setelah itu terdapat bagian *data collection* yang mengenai sumber data yang akan digunakan pada penelitian ini seperti halnya sumber data, jumlah data dan pembagian data untuk training dan testing.

3.1 Data Collection

Data yang penulis ambil untuk penelitian ini merupakan data citra satelit suatu daerah seusai mengalami bencana alam. Dataset tersebut didapatkan oleh penulis melalui website public penyedia dataset yaitu *xView2 : Assess Building Damage*. Pada penelitian ini, penulis akan membagi data tersebut menjadi komposisi 70:30. Artinya akan ada 70% data untuk training dan ada 30% data untuk testing.

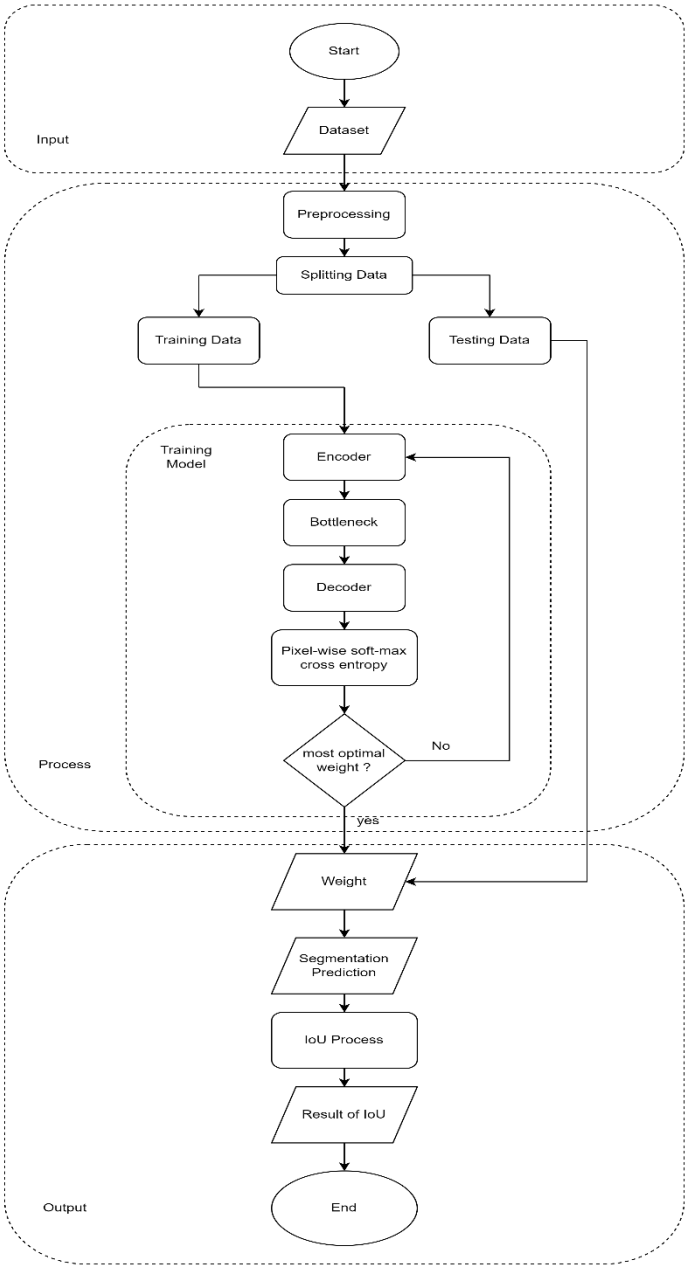
Jumlah kelas yang ada pada data yang sudah didapatkan terdapat 4 kelas, yaitu *no damage, minor damage, major damage, dan destroyed*. Kelas ini dapat dilihat pada gambar *ground truth* yang telah didapatkan. Namun untuk penelitian ini, penulis akan menambahkan satu kelas lagi yaitu kelas *unlabelled*. Sehingga total kelas yang digunakan pada penelitian ini berjumlah 5 kelas.

Tabel 3.1 Label Kelas

Kode Kelas	Deskripsi	Warna
0	*No damage*	
1	*Minor Damage*	
2	*Major Damage*	
3	*Destroyed*	
4	*Unlabelled*	

3.2 Desain Sistem

Pada penelitian ini, penulis sudah menyusun desain sistem yang akan menggambarkan sistem dari penelitian ini. Isi dari desain sistem ini akan berisi alur mulai dari *input data*, *preprocessing*, *main process*, hingga *output*. Desain sistem untuk penelitian ini dapat dilihat pada gambar 3.1.



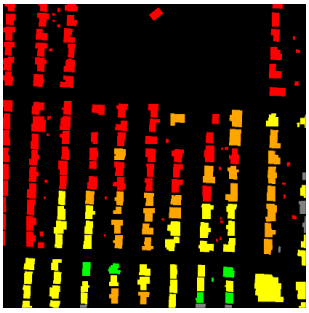
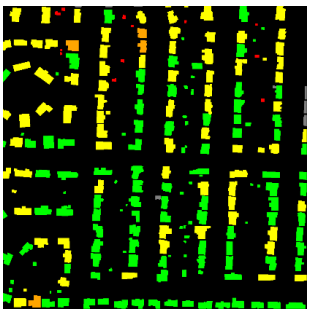
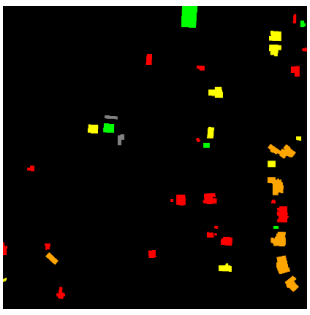
Gambar 3.1 Desain sistem

3.2.1 Input Data

Pada proses ini bisa dikatakan sebagai proses inisialisasi data yaitu proses inisialisasi data ke dalam sistem. Data yang diinisialisasikan merupakan data citra satelit daerah pasca bencana alam. Data citra satelit akan berjumlah sebanyak 42

gambar dan data ground truth sebanyak 42 gambar. Citra tersebut berformat gambar *Portable Network Graphics* (*.png).

Tabel 3.2 Contoh data citra satelit beserta *ground truth*

No	Citra satelit daerah pasca bencana alam	*Ground truth*
1.		
2.		
3.		

3.2.2 Preprocessing

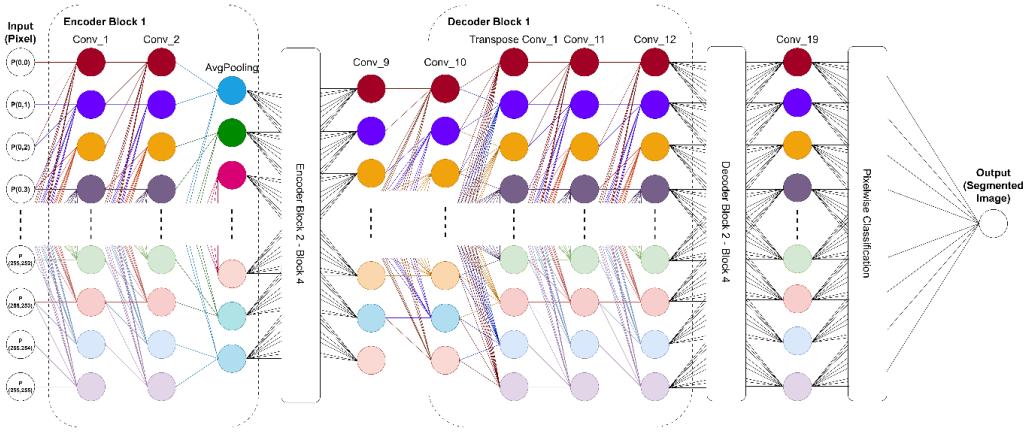
Gambar 3.2 Alur prepocessing

Proses selanjutnya yaitu tahapan *preprocessing* data yang dilakukan sebelum memasuki proses *training*. Hal yang akan dilakukan pada tahap ini yaitu melakukan *patchify* pada gambar lalu melakukan normalisasi hasil *patchify*. *Patchify* pada gambar yaitu membagi gambar ke dalam ukuran yang kecil – kecil. Hal ini dilakukan karena ukuran resolusi gambar asli yang berukuran 1024 x 1024 terlalu besar jika langsung digunakan untuk proses konvolusi dan *training*, oleh karena itu saya akan memecahnya menjadi masing masing berukuran 256 x 256.

Setelah dilakukan proses *patchify*, dilanjutkan dengan melakukan proses augmentasi yang bertujuan untuk lebih memperkaya data dan mengurangi terjadinya overfitting saat proses pelatihan. Augmentasi yang dilakukan disini yaitu melakukan rotasi 90°, 180°, 270° pada setiap gambar yang sudah dilakukan *patchify*.

Setelah melakukan augmentasi, akan dilakukan proses normalisasi. Proses ini akan membagi nilai dari setiap pixel dengan 255, sehingga nilai tiap pixel hanya akan bernilai 0 – 1. Hal ini bertujuan untuk mengurangi beban komputasi saat melakukan proses pelatihan.

3.2.3 Model Arsitektur



Gambar 3.3 Arsitektur dalam bentuk topologi

Model arsitektur pada penelitian ini menggunakan arsitektur dengan 4 pasang blok *encoder-decoder* dan satu blok *bottleneck*. Pada setiap blok *encoder* akan dilakukan konvolusi dengan aktivasi ReLU lalu setelah itu akan dilakukan *batch normalization* dan akhirya akan dilakukan *pooling*. Selanjutnya bagian *decoder* akan dilakukan *transpose convolution* lalu juga akan melakukan *concatenate* dengan memanfaatkan *feature map* pada *encoder*. Setelah melakukan *transpose convolution*, maka akan dilakukan konvolusi seperti biasanya dengan melakukan *batch normalization* juga setelahnya. Diantara *encoder-decoder* terdapat 1 blok *bottleneck* yang akan melakukan konvolusi tanpa melakukan *pooling*. Pada akhir bagian setelah *decoder* akan dilakukan proses *pixelwise* untuk mengklasifikasi tiap pixelnya berdasarkan kelas yang telah ditentukan dan juga bertujuan untuk perhitungan *loss function*.

3.2.3.1 Encoder

Encoder merupakan salah satu bagian pada arsitektur yang bertujuan untuk melakukan ekstrasi fitur pada citra. Citra yang telah melalui tahapan *preprocessing* akan diproses kembali oleh blok encoder untuk menghasilkan *feature map*.Pada setiap blok *encoder* dari arsitektur memiliki sistem kerja yang serupa yaitu melakukan konvolusi gambar *input*, lalu hasil dari konvolusi tersebut akan dilakukan pooling sehingga menghasilkan *feature map*. Selanjutnya *feature map* akan diteruskan ke blok selanjutnya untuk dilakukan cara kerja yang sama. Berikut adalah *pseudocode* dari *encoder* :

```
function convolution(input, kernel, bias, stride):
    // Mengambil dimensi dari input dan kernel
    batch_size, input_height, input_width, input_channels =
shape(input)
    kernel_height, kernel_width, _, output_channels =
shape(kernel)

    // Menghitung dimensi dari output
    output_height = (input_height - kernel_height) // stride +
1
    output_width = (input_width - kernel_width) // stride + 1

    // Membuat output map yang kosong dengan dimensi yang telah
dihitung
    output = zeros((batch_size, output_height, output_width,
output_channels))

    // Perulangan untuk melakukan konvolusi
    for b in range(batch_size):
        for y_out in range(output_height):
            for x_out in range(output_width):
                for c_out in range(output_channels):
                    // Menghitung batas-batas patch pada input
                    y_in_start = y_out * stride
                    y_in_end = y_in_start + kernel_height
                    x_in_start = x_out * stride
                    x_in_end = x_in_start + kernel_width

                    // Mengambil patch dari input menggunakan
batas-batas yang telah dihitung
                    input_patch = input[b, y_in_start:y_in_end,
x_in_start:x_in_end, :]

                    // Melakukan konvolusi antara input patch
dan kernel
                    output[b, y_out, x_out, c_out] =
sum(input_patch * kernel[:, :, :, c_out])

                // Menambahkan nilai bias ke setiap titik
output
                output[b, y_out, x_out, :] += bias

    // Mengembalikan output map yang dihasilkan
    return output
```

Lanjutan *pseudocode*

```
function average_pooling(input, pool_size, stride):
    // Mengambil dimensi dari input
    batch_size, input_height, input_width, input_channels =
shape(input)

    // Menghitung dimensi dari output
    output_height = (input_height - pool_size) // stride + 1
    output_width = (input_width - pool_size) // stride + 1

    // Membuat output map yang kosong dengan dimensi yang telah
dihitung
    output = zeros((batch_size, output_height, output_width,
input_channels))

    // Perulangan untuk melakukan average pooling
    for b in range(batch_size):
        for y_out in range(output_height):
            for x_out in range(output_width):
                for c in range(input_channels):
                    // Menghitung batas-batas patch pada input
                    y_in_start = y_out * stride
                    y_in_end = y_in_start + pool_size
                    x_in_start = x_out * stride
                    x_in_end = x_in_start + pool_size

                    // Mengambil patch dari input menggunakan
batas-batas yang telah dihitung
                    input_patch = input[b, y_in_start:y_in_end,
x_in_start:x_in_end, c]

                    // Menghitung rata-rata nilai dari patch
dan menyimpannya pada output map
                    output[b, y_out, x_out, c] =
mean(input_patch)

    // Mengembalikan output map yang dihasilkan
    return output
```

3.2.3.1.1 Convolution Layer

Proses ini yang akan dikerjakan pertama kali pada tiap blok *encoder*, rumus dari konvolusi ialah (Goodfellow, Bengio, and Courville n.d. , 2016) :

$$C(i,j) = \sum_{m}^{1} \sum_{n}^{1} I(i+m, j+n) * K(m,n) \qquad (3.1)$$

Keterangan :
(i, j) = indeks piksel.
(m, n) = indeks kernel.
K = kernel.
C = hasil konvolusi.
I = citra input.

Contoh :

Matrik citra 3 *channel RGB* 6 x 6 :

$$\begin{bmatrix} (68,87,108) & (86,108,110) & (40,67,53) & (61,77,76) & (45,68,62) & (50,66,67) \\ (35,62,40) & (75,93,99) & (60,80,94) & (127,139,138) & (47,83,71) & (105,115,118) \\ (71,88,108) & (26,49,46) & (63,77,78) & (35,71,52) & (44,74,59) & (52,71,67) \\ (37,50,47) & (62,95,110) & (97,114,123) & (32,55,44) & (71,89,88) & (69,85,83) \\ (30,50,37) & (49,65,85) & (98,112,128) & (25,47,39) & (37,62,56) & (29,51,44) \\ (15,31,26) & (21,41,34) & (32,49,42) & (41,56,56) & (33,60,47) & (39,59,55) \end{bmatrix}$$

Kernel 3 x 3 :

$$\begin{bmatrix} -0.3 & 0.21 & -0.07 \\ 0.19 & 0.1 & -0.01 \\ -0.04 & -0.02 & 0.08 \end{bmatrix}$$

Hasil konvolusi *channel red* :

$$\begin{bmatrix} 2 & 34 & 10 & 16 \\ -5 & 18 & -9 & 11 \\ 5 & 24 & 5 & 16 \\ 14 & 8 & 9 & 0 \end{bmatrix}$$

Hasil konvolusi *channel green* :

$$\begin{bmatrix} 7 & 29 & 17 & 20 \\ 0 & 21 & -2 & 14 \\ 10 & 25 & 6 & 20 \\ 16 & 13 & 9 & 5 \end{bmatrix}$$

Hasil konvolusi *channel blue* :

$$\begin{bmatrix} 10 & 37 & 14 & 19 \\ -1 & 22 & -8 & 13 \\ 15 & 28 & 5 & 20 \\ 21 & 13 & 13 & 1 \end{bmatrix}$$

3.2.3.1.2 Aktivasi ReLU

Setiap selesai dilakukan konvolusi, maka akan dilakukan aktivasi dengan menggunakan ReLu, berikut rumus perhitungannya :

$$f(x) = \max(0, x) \tag{3.2}$$

Keterangan :
F(x) = hasil dari fungsi ReLU.
X = input ke fungsi ReLU.

Hasil dari proses konvolusi sebelumnya akan dilakukan aktivasi ReLu, berikut penerapan ReLU pada hasil konvolusi sebelumnya :

Hasil aktivasi ReLU *channel red* :

$$\begin{bmatrix} 2 & 34 & 10 & 16 \\ -5 & 18 & -9 & 11 \\ 5 & 24 & 5 & 16 \\ 14 & 8 & 9 & 0 \end{bmatrix} \text{ReLU} \begin{bmatrix} 2 & 34 & 10 & 16 \\ 0 & 18 & 0 & 11 \\ 5 & 24 & 5 & 16 \\ 14 & 8 & 9 & 0 \end{bmatrix}$$

Hasil aktivasi ReLU *channel green* :

$$\begin{bmatrix} 7 & 29 & 17 & 20 \\ 0 & 21 & -2 & 14 \\ 10 & 25 & 6 & 20 \\ 16 & 13 & 9 & 5 \end{bmatrix} \text{ReLU} \begin{bmatrix} 7 & 29 & 17 & 20 \\ 0 & 21 & 0 & 14 \\ 10 & 25 & 6 & 20 \\ 16 & 13 & 9 & 5 \end{bmatrix}$$

Hasil aktivasi ReLU *channel blue* :

$$\begin{bmatrix} 10 & 37 & 14 & 19 \\ 0 & 22 & 0 & 13 \\ 15 & 28 & 5 & 20 \\ 21 & 13 & 13 & 1 \end{bmatrix} \text{ReLU} \begin{bmatrix} 10 & 37 & 14 & 19 \\ 0 & 22 & 0 & 13 \\ 15 & 28 & 5 & 20 \\ 21 & 13 & 13 & 1 \end{bmatrix}$$

3.2.3.1.3 Pooling Layer

Proses Selanjutnya gambar *input* yang sudah dilakukan konvolusi, hasilnya akan dilakukan *pooling*. Pada penelitian ini akan menggunakan *average pooling*. Berikut adalah rumusnya :

$$P(i,j) = \frac{1}{F^2} \sum_{x=i}^{i+F-1} \sum_{y=j}^{j+F-1} I(x,y) \quad (3.3)$$

Keterangan :
P(i,j) = nilai yang dihasilkan oleh operasi *Average pooling* pada posisi (i,j) dalam gambar.
F = ukuran filter.
I(x,y) = nilai piksel pada posisi (x,y) dalam gambar asli sebelum pooling.

Berikut adalah proses *pooling* 2x2 pada setiap *channel* setelah dilakukan aktivasi ReLU :

Hasil *pooling channel red* :

$$\begin{bmatrix} 2 & 34 & 10 & 16 \\ 0 & 18 & 0 & 11 \\ 5 & 24 & 5 & 16 \\ 14 & 8 & 9 & 0 \end{bmatrix} \textit{Average pooling} \begin{bmatrix} 13 & 9 \\ 12 & 7 \end{bmatrix}$$

Hasil *pooling channel green* :

$$\begin{bmatrix} 7 & 29 & 17 & 20 \\ 0 & 21 & 0 & 14 \\ 10 & 25 & 6 & 20 \\ 16 & 13 & 9 & 5 \end{bmatrix} \textit{Average pooling} \begin{bmatrix} 14 & 12 \\ 16 & 10 \end{bmatrix}$$

Hasil *pooling channel blue* :

$$\begin{bmatrix} 10 & 37 & 14 & 19 \\ 0 & 22 & 0 & 13 \\ 15 & 28 & 5 & 20 \\ 21 & 13 & 13 & 1 \end{bmatrix} \textit{Average pooling} \begin{bmatrix} 17 & 11 \\ 19 & 9 \end{bmatrix}$$

Setelah *pooling* berhasil dilakukan, maka *feature map* akan diteruskan ke blok selanjutnya dengan nilai filter yang akan dipangkatkan. Proses ini dinamakan sebagai proses *downsampling.*

3.2.3.2 Bottleneck

Pada proses ini, fitur – fitur yang telah didapatkan dari *encoder* akan dilakukan proses konvolusi seperti biasa namun bedanya adalah dengan tidak melakukan *pooling*. *Feature map* dari blok terakhir encoder akan memasuki blok *bottleneck*, proses yang akan terjadi yaitu akan melakukan dua kali proses konvolusi. Setelah Fungsi dari bottleneck sendiri ialah supaya mempertahankan informasi – informasi dari gambar tersebut.

3.2.3.3 Decoder

Proses Selanjutnya yaitu decoder yang akan melakukan proses *transpose convolution* pada *feature map* yang telah di hasilkan oleh blok *bottleneck*. Proses perblok dari *decoder* akan terjadi satu kali proses *transpose convolution* dan dua kali proses *convolution* tanpa *pooling.* Pada *decoder* juga akan terjadi proses *concatenate* yaitu penggabungan antara *feature map* dari *encoder* dengan hasil *transpose convolution* pada blok *decoder* yang sejajar dengannya. Berikut adalah *pseudocode* dari proses *decoder* :

```
function transpose_convolution(input, kernel, stride):
    // Mengambil dimensi dari input dan kernel
    input_shape = get_shape(input)
    kernel_shape = get_shape(kernel)

    // Menghitung dimensi dari output
    output_height = (input_shape.height - 1) * stride +
kernel_shape.height
    output_width = (input_shape.width - 1) * stride +
kernel_shape.width

    // Membuat output map yang kosong dengan dimensi yang telah
dihitung
    output = zeros(output_height, output_width,
kernel_shape.depth)

    // Perulangan untuk melakukan transposed convolution
    for k_channel in range(kernel_shape.depth):
        for out_row in range(output_height):
            for out_col in range(output_width):
                in_row = out_row * stride
                in_col = out_col * stride

                // Perulangan untuk mengalikan input patch
dengan kernel
                for k_row in range(kernel_shape.height):
                    for k_col in range(kernel_shape.width):
                        in_pixel_row = in_row + k_row
                        in_pixel_col = in_col + k_col

                        // Melakukan transposed convolution dan
menyimpan hasilnya pada output map
                        output[out_row][out_col][k_channel] +=
input[in_pixel_row][in_pixel_col][k_channel] *
kernel[k_row][k_col][k_channel]

    // Mengembalikan output map yang dihasilkan
    return output
```

Lanjutan *psuedocode*

```
function concatenate(inputs1, inputs2, axis):
    // Mengambil dimensi dari kedua input arrays
    shape1 = get_shape(inputs1)
    shape2 = get_shape(inputs2)

    // Memeriksa apakah dimensi yang ditentukan untuk
concatenation axis sesuai
    if shape1[axis] != shape2[axis]:
        throw "Dimensi tidak sesuai untuk concatenation axis"

// Menghitung dimensi dari output
    result_shape = copy(shape1)
    result_shape[axis] += shape2[axis]

    // Membuat output array yang kosong dengan dimensi yang
telah dihitung
    result = zeros(result_shape)

    // Melakukan concatenation untuk input arrays pertama
    for i in range(shape1[axis]):
        indices1 = create_indices(shape1)
        indices1[axis] = i
        result[indices1] = inputs1[indices1]

    // Melakukan concatenation untuk input arrays kedua
    for i in range(shape2[axis]):
        indices2 = create_indices(shape2)
        indices2[axis] = i
        result[indices2[axis] + shape1[axis]] =
inputs2[indices2]

    // Mengembalikan hasil concatenation
    return result

function upsampling_block(inputs, skip_connections, filters,
kernel_size, stride):
    // Melakukan transposed convolution pada input
    x = transpose_convolution(inputs, filters, stride)

    // Melakukan concatenation antara hasil transposed
convolution dan skip connections
    x = concatenate(inputs1=x, inputs2=skip_connections, axis=-
1)
    // Melakukan convolution block pada hasil concatenation
    x = convolution_block(x, filters)
    x = convolution_block(x, filters)

    // Mengembalikan hasil upsampling block
    return x
```

Lanjutan *psuedocode*

```
// Melakukan proses upsampling untuk decoder blocks dengan
menggabungkan hasil upsampling block dengan skip connections
decoder_block4 = upsampling_block(bottleneck, encoder_block3,
filters=512)
decoder_block3 = upsampling_block(decoder_block4,
encoder_block2, filters=256)
decoder_block2 = upsampling_block(decoder_block3,
encoder_block1, filters=128)
decoder_block1 = upsampling_block(decoder_block2, inputs,
filters=64)

// Melakukan convolution block pada hasil decoder blocks
terakhir
decoder_output = convolution_block(decoder_block1,
filters=num_classes, activation='softmax')
```

Berikut adalah rumus *transpose convolution* pada decoder :

$$C(i,j) = \sum_{m}^{1} \sum_{n}^{1} I(i-m, j-n) * K(m,n) \qquad (3.4)$$

Keterangan :
(i, j) = indeks piksel.
(m, n) = indeks kernel.
K = kernel.
C = hasil konvolusi.
I = citra input.

Berikut adalah contoh dari perhitungan *transpose convolution* dari hasil *pooling* di atas :

Channel red :

$\begin{bmatrix} 13 & 9 \\ 12 & 7 \end{bmatrix}$ Transpose convolution dengan kernel $\begin{bmatrix} -0.3 & 0.21 & -0.07 \\ 0.19 & 0.1 & -0.01 \\ -0.04 & -0.02 & 0.08 \end{bmatrix}$

$$\begin{bmatrix} -4 & 2 & -0.9 & 0 \\ 2 & 1 & -0.1 & 0 \\ -0.5 & -0.2 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -2 & 1.9 & -0.6 \\ 0 & 1.7 & 0.9 & -0.1 \\ 0 & -0.3 & -0.2 & 0.7 \\ 0 & 0 & 0 & 0 \end{bmatrix} +$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -3,6 & 2.5 & -0.8 & 0 \\ 2.2 & 1.2 & -0.12 & 0 \\ -0.4 & -0.2 & 0.9 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2.1 & 1.4 & -0.49 \\ 0 & 1.3 & 0.7 & -0.1 \\ 0 & -0.28 & -0.1 & 0.5 \end{bmatrix}$$

$$= \begin{bmatrix} -4 & -2 & 1 & -0.6 \\ 3.6 & 5.5 & 0.36 & -0.21 \\ 1.7 & 1.3 & 0.47 & 0.5 \\ -0.4 & -0.68 & 0.8 & 0.5 \end{bmatrix} \text{ReLU} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 3.6 & 5.5 & 0.36 & 0 \\ 1.7 & 1.3 & 0.47 & 0.5 \\ 0 & 0 & 0.8 & 0.5 \end{bmatrix}$$

Channel green :

$\begin{bmatrix} 14 & 12 \\ 16 & 10 \end{bmatrix}$ Transpose convolution dengan kernel $\begin{bmatrix} -0.3 & 0.21 & -0.07 \\ 0.19 & 0.1 & -0.01 \\ -0.04 & -0.02 & 0.08 \end{bmatrix}$

$$\begin{bmatrix} -4.2 & 2.9 & -1 & 0 \\ 2.6 & 1.4 & -0.1 & 0 \\ -0.6 & -0.3 & 1.1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -3.6 & 2.5 & -0.8 \\ 0 & 2.2 & 1.2 & -0.12 \\ 0 & -0.48 & -0.24 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} +$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -4.8 & 3.3 & -1.1 & 0 \\ 3 & 1.6 & -0.16 & 0 \\ -0.64 & -0.32 & 1.2 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -3 & 2.1 & -0.7 \\ 0 & 1.9 & 1 & -0.1 \\ 0 & -0.4 & -0.2 & 0.8 \end{bmatrix}$$

$$= \begin{bmatrix} -4.2 & -0.7 & 1.4 & -0.8 \\ -1.2 & 1.7 & 0.4 & -0.1 \\ 2.4 & 2.7 & 1.7 & 0.9 \\ -0.64 & 0.6 & 0.48 & 0.8 \end{bmatrix} \text{ReLU} \begin{bmatrix} 0 & 0 & 1.4 & 0 \\ 0 & 1.7 & 0.4 & 0 \\ 2.4 & 2.7 & 1.7 & 0.9 \\ 0 & 0.6 & 0.48 & 0.8 \end{bmatrix}$$

Channel blue :

$\begin{bmatrix} 17 & 11 \\ 19 & 9 \end{bmatrix}$ Transpose convolution dengan kernel $\begin{bmatrix} -0.3 & 0.21 & -0.07 \\ 0.19 & 0.1 & -0.01 \\ -0.04 & -0.02 & 0.08 \end{bmatrix}$

$$\begin{bmatrix} -5.1 & 3.5 & -1.1 & 0 \\ 3.2 & 1.7 & -0.17 & 0 \\ -0.68 & -0.34 & 1.3 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -3.3 & 2.3 & -0.77 \\ 0 & 2.1 & 1.1 & -0.11 \\ 0 & -0.44 & -0.22 & 0.88 \\ 0 & 0 & 0 & 0 \end{bmatrix} +$$

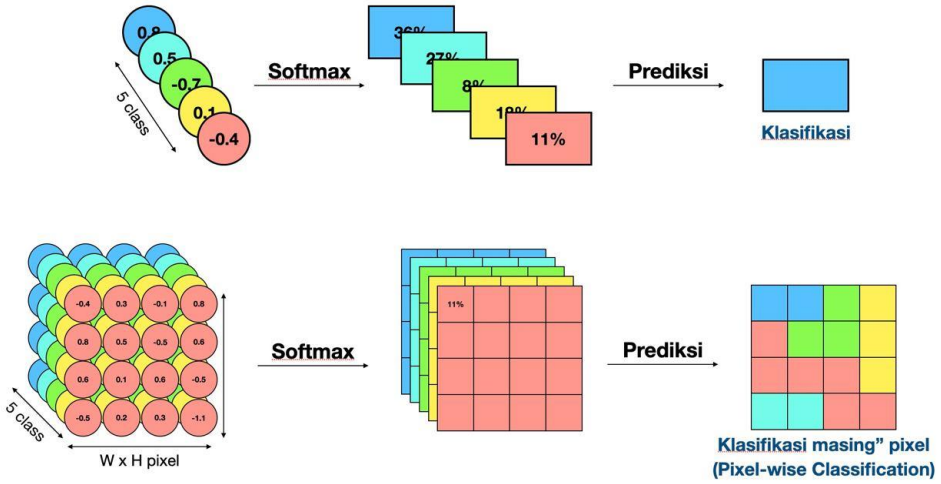
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ -5.7 & 4 & -1.3 & 0 \\ 3.6 & 2 & -0.2 & 0 \\ -0.7 & -0.4 & 1.5 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -2.7 & 1.9 & -0.6 \\ 0 & 1.7 & 0.9 & -0.09 \\ 0 & -0.36 & -0.2 & 0.72 \end{bmatrix}$$

$$= \begin{bmatrix} -5.1 & 0.2 & 1.2 & -0.77 \\ -2.5 & 3.2 & -0.28 & -0.6 \\ 2.9 & 1.9 & 0.91 & 0.8 \\ -0.7 & 1.3 & 2.4 & 0.72 \end{bmatrix} \text{ReLU} \begin{bmatrix} 0 & 0.2 & 1.2 & 0 \\ 0 & 3.2 & 0 & 0 \\ 2.9 & 1.9 & 0.91 & 0.8 \\ 0 & 1.3 & 2.4 & 0.72 \end{bmatrix}$$

Setelah itu akan dilakukan concatenate yaitu penggabungan fitur – fitur pada setiap blok yang diambil dari blok encoder dan digabungkan ke blok decoder yang sejajar melalui skip connections. Hal ini akan membantu proses up-scaling menjadi lebih mudah dalam mengembalikan resolusi gambar menjadi seperti semula.

3.2.3.4 Pixelwise

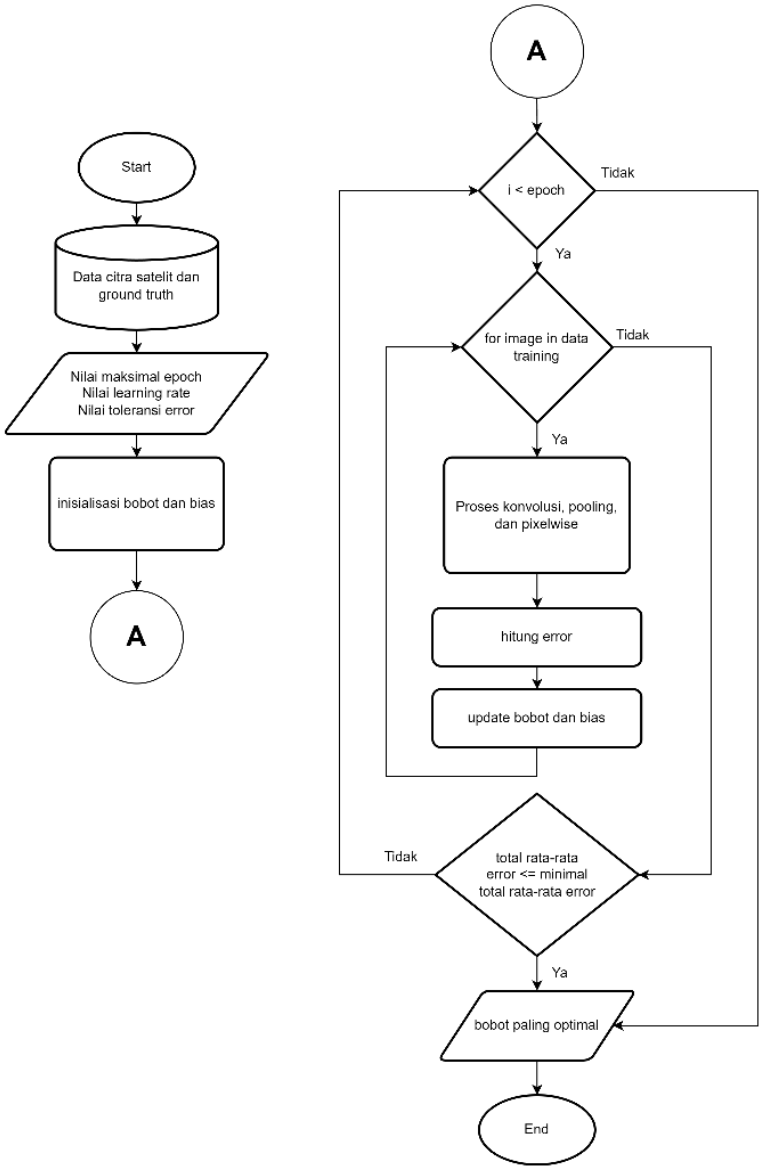
Setelah gambar telah melewati blok *decoder* terakhir, langkah terakhir yaitu melakukan proses yang dinamakan *pixelwise*. Proses ini akan me klasifikasikan per pixel gambar dengan menggunakan aktivasi *softmax* untuk menentukan nilai mana yang paling besar pada setiap kelas yang telah ditentukan. Proses ini akan digabungkan dengan *cross entropy* yaitu nantinya perhitungan *loss function* akan dilakukan per *pixel* untuk dilihat nilainya. Jika ternyata nilai dari *loss function* tidak begitu baik maka akan dilakukan proses *backpropagation* untuk memperbarui nilai bobot pada *filter*. Gambar 3.5 diambil dan dimodifikasi dari (Strohmann *et al*., 2019)



Gambar 3.4 Pixelwise

3.2.4 Training Model Convoutional Neural Network

Untuk bisa mendapatkan model yang mampu melakukan segmentasi dengan akurat, maka langkah berkutnya yaitu melakukan proses *training*. Proses training bertujuan agar model *convolutional neural network* memiliki nilai bobot yang paling optimal. Sehingga dengan nilai bobot yang paling optimal mampu melakukan prediksi untuk segmentasi citra yang diberikan. Proses *training* akan menggunakan gambar citra aerial daerah pasca bencana alam dan citra *ground truth* yang telah diinisialisasi di awal. *Flowchart* dari proses *training* diberikan pada gambar 3.6 yang diambil dan dimodifikasi dari (Ojha *et al*., 2012)



Gambar 3.5 Alur training

Setelah memanggil data inputan, pada proses training perlu dilakukan juga inisialisasi parameter yang akan digunakan dalam pelatihan. Parameter tersebut biasa disebut dengan *hyperparameter*. Diantara *hyperparameter* tersebut ialah *learning rate*, *epoch*, *batch size,* dan toleransi *error*. Setelah di inisialisasi, nilai dari *hyperparameter* akan bersifat konstanta atau tidak akan berubah – ubah

selama proses pelatihan. Berikut adalah tabel definisi dari setiap *hyperparameter* di atas.

Tabel 3.3 Definisi *Hyperparameter*

No	*Hyperparameter*	Definisi
1.	*Learning rate*	*Learning rate* adalah parameter yang menentukan seberapa besar langkah yang diambil selama proses optimasi pada model *machine learning*. Penelitian ini akan menggunakan 0.001 untuk *learning rate*
2.	*Epoch*	*Epoch* adalah satu kali putaran lengkap melalui seluruh dataset pelatihan selama pelatihan model *machine learning*. Penelitian ini akan menggunakan 50 epoch
3.	*Batch size*	*Batch size* adalah jumlah contoh pelatihan yang digunakan dalam satu iterasi pada proses pelatihan model. Penelitian ini akan menggunakan 2 *batch size*
4.	Toleransi *error*	Toleransi *error* mengacu pada tingkat kesalahan yang dapat diterima atau batas kesalahan yang diizinkan selama pelatihan model, sering digunakan untuk menghentikan pelatihan jika tingkat kesalahan telah mencapai batas tertentu.

Setelah melakukan inisialisasi *hyperparameter*, maka selanjutnya proses pelatihan model akan berlangsung. Proses pelatihan terbagi menjadi dua tahap yaitu propagasi maju dan propagasi mundur. Propagasi maju adalah proses dimana model melakukan ekstrasi fitur dari input dan melakukan pengecekan error pada akhir proses dengan bobot yang sudah didapatkan. Seperti yang saya sebutkan sebelumnya, bahwa proses pelatihan nilai bobot akan terus diperbarui hingga ke titik paling. Memperbarui nilai bobot akan terjadi pada proses propagasi mundur dimana semua bobot akan di perbarui. Setelah diperbarui maka propagasi maju akan dijalankan kembali. Hal ini akan terus dijalankan hingga menghasilkan bobot yang paling optimal.

3.2.4.1 Propagasi Maju

Pada propagasi maju, data input akan melalui 4 blok encoder, 1 blok bottleneck dan 1 blok deccoder. Pada setiap blok encoder akan dilakukan dua kali proses konvolusi, satu kali proses aktivasi dengan ReLU dan satu kali proses pooling.

$$\mathrm{C}(i,j)^{(z)} = \sum_{m}^{1}\sum_{n}^{1} I(i+m, j+n) * K(m,n) \tag{3.5}$$

Keterangan :
(i, j) = indeks piksel.
(m, n) = indeks kernel.
K = kernel.
C = hasil konvolusi.
I = citra input.
z = iterasi konvolusi dimana z <= 28.

Setelah mendapatkan hasil konvolusi yaitu $\mathrm{C}(i,j)^{(z)}$ maka selanjutnya hasil tersebut akan dilakukan aktivasi ReLU dengan persamaan 3.6.

$$f\big(\mathrm{C}(i,j)^{(z)}\big)^{(v)} = \max\big(0, \mathrm{C}(i,j)^{(z)}\big) \tag{3.6}$$

Keterangan :
f(x) = hasil dari fungsi ReLU dimana merupakan inputan yang berupa hasil konvolusi.
v = iterasi ReLU dimana v <= 8.

Setelah itu dilakukan proses konvolusi dan proses ReLU satu kali lagi sehingga menghasilkan nilai $f\big(\mathrm{C}(i,j)^{(z)}\big)^{(v)}$ akan di inisialisasikan $R^{(v)}$ dimana R merupakan hasil dari ReLU dengan v merupakan iterasinya. Setelah itu akan dilakukan proses *pooling* yang dapat dilihat pada persamaan 3.7.

$$AvgPooling(R^{(v)})^{(b)} = \frac{1}{F^2}\sum_{x=i}^{i+F-1}\sum_{y=j}^{j+F-1} I(x,y) \tag{3.7}$$

Keterangan :
P(i,j) = nilai yang dihasilkan oleh operasi *Average pooling* pada posisi (i,j) dalam gambar.
F = ukuran filter.
I(x,y) = nilai piksel pada posisi (x,y) dala gambar asli sebelum pooling.
b = iterasi pooling dimana b <= 4.

Semua proses propagasi maju di atas merupakan proses propagasi maju pada blok *encoder*. Selanjutnya hasil pooling dari blok keempat *encoder* akan dilakukan proses konvolusi dua kali yaitu pada blok *bottleneck* sebelum memasuki blok decoder. Setelah konvolusi pada bottleneck berhasil didapatkan maka selanjutnya akan dilakukan proses *transpose convolution* pada blok *decoder* guna mengembalikan kembali dimensi dari data input, persamaan dari proses propagasi maju *transpose convolution* pada *decoder* dapat dilihat pada persamaan 3.8.

$$T(i,j)^{(z)} = \sum_{m}^{1} \sum_{n}^{1} I(i-m, j-n) * K(m,n) \tag{3.8}$$

Keterangan :
(i, j) = indeks piksel.
(m, n) = indeks kernel.
K = kernel.
C = hasil konvolusi.
I = citra input.
z = iterasi konvolusi dimana z <= 28.

Hasil dari *transpose convolution* $T(i,j)^{(z)}$ akan dilakukan proses konvolusi lagi sebanyak dua kali. Dengan demikian proses pada satu blok *decoder* sudah dijalankan, selanjutnya proses serupa akan dilakukan hingga blok keempat.

Hasil dari blok decoder keempat $T(i,j)^{(z)}$ selanjutnya akan dilakukan proses konvolusi dengan *kernel* 1 x 1 yang mana hal ini terjadi pada *layer pixelwise.* Berikut adalah persamaan pada layer pixelwise.

$$P^{\square} = \sum_{m}^{1} \sum_{n}^{1} I(i+m, j+n) * K(m,n) \tag{3.9}$$

Keterangan :
(i, j) = indeks piksel.
(m, n) = indeks kernel.
K = kernel 1 x 1.
C = hasil konvolusi.
I = citra input.
p = hasil dari konvolusi 1 x 1 pada layer pixelwise.

Selanjutnya hasil dari konvolusi *layer pixelwise* akan dioperasikan dengan fungsi *softmax* untuk mendapatkan probabilitas piksel tersegmentasi dari setiap kelas. Persamaan dari *softmax* dapat dilihat pada persamaan 3.10.

$$\sigma(p)^{\square} = \frac{e^{P_i}}{\sum_{j=0}^{L} e^{P_j}} \tag{3.10}$$

Keterangan :
p_i = elemen ke – i dari vektor hasil konvolusi $p_{\square}$
$\sum_{j=0}^{L} e^{P_j}$ = jumlah dari eksponen semua elemen vektor p
L = jumlah kelas.

Setelah melewati *layer pixelwise*, maka langkah terkahir yaitu melakukan perhitungan error dengan menggunakan *categorical crossentropy*. Persamaannya dapat dilihat pada persamaan 3.11.

$$Categorical\ Crossentropy = -\sum_{i}^{\square} y_i * \log(\hat{y}_i) \tag{3.11}$$

Keterangan :
i = indeks dari setiap kelas atau kategori.
y_i = elemen ke - i dari vektor yang menyatakan distribusi probabilitas sebenarnya (label yang diharapkan, mungkin dalam bentuk one-hot encoding).
$\hat{y}_i$ = elemen ke - i dari vektor probabilitas prediksi yang dihasilkan oleh model.

3.2.4.2 Propagasi Mundur

Proses propagasi mundur merupakan langkah krusial dalam pelatihan model segmentasi gambar. Pada tahap ini, nilai *error* diarahkan mundur ke

seluruh lapisan jaringan untuk mengupdate bobot yang terdapat pada setiap kernel konvolusi. Tujuan utama adalah memperbarui bobot agar hasil prediksi semakin mendekati nilai sebenarnya pada setiap piksel citra.

Perhitungan error, diwakili oleh $\delta z(i)$, dihitung sebagai selisih antara nilai prediksi ($z(i)$) dan nilai sebenarnya ($y(i)$) pada setiap piksel citra. Persamaannya dapat dirumuskan pada persamaan 3.12:

$$\delta z(i) = z(i) - y(i) \tag{3.12}$$

Setelah mendapatkan nilai *error*, langkah selanjutnya adalah memperbarui bobot w baru dan bias baru pada kernel konvolusi ke i. Hal ini dilakukan dengan menggunakan metode *Stochastic Gradient Descent* (SGD). Persamaannya dapat disederhanakan sebagai berikut:

$$wbaru(i) = wlama(i) + \alpha * \delta z(i) * z(i-1)(i) \tag{3.13}$$

$$bbaru(i) = blama(i) + \alpha * \delta z(i) \tag{3.14}$$

Keterangan :

$wbaru(i)$ dan $bbaru(i)$	= bobot dan bias yang diperbarui pada kernel konvolusi ke i
$wlama(i)$ dan $blama(i)$	= bobot dan bias sebelum diperbarui pada kernel konvolusi ke i
α	= tingkat pembelajaran yang mengatur seberapa besar langkah pembelajaran model.
$\delta z(i)$	= nilai error pada output.
$z(i-1)$	= output dari layer sebelumnya.

3.2.5 Evaluation

Setelah model memberikan hasil segmentasi citra, langkah selanjutnya yaitu perhitungan evaluasi model terhadap hasil yang sudah diberikan. Evaluasi model pada penelitian ini akan menggunakan *Mean Intersection Over the Union Score* atau biasa disingkat MIoU. Nilai yang dihasilkan akan mempunyai rentang

0 – 1 dimana semakin mendekati angka 1 maka bentuk hasil prediksi akan semakin mendekati ground truth (M. Z. Khan *et al*., 2021). Rumus perhitungan dapat dilihat pada persamaan 3.15 :

$$MIoU = \sum_{1}^{n} \frac{TP}{(TP + FP + FN)} * \frac{1}{n} \quad (3.15)$$

Keterangan :
TP = nilai true positive
FP = nilai false positive
FN = nilai false negative

3.3 Experiment

Pada penelitian ini, penulis akan menjabarkan apa saja skenario percobaan yang akan dilakukan pada penelitian ini. Hal ini dilakukan agar peneliti mengerti pada skenario mana, model memiliki bobot paling optimal dan akurasi paling bagus.

Skenario pertama adalah skenario dalam penggunaan *pooling*. Dalam arsitektur yang peneliti ajukan, proses *pooling* dilakukan dengan *average pooling*. Namun penggunaan metode *pooling* lain juga akan digunakan pada penilitan ini dengan dicantumkan pada skenario uji coba.

Tabel 3.4 Skenario Pertama

Kode Skenario	***Pooling Layer***
A	*Average pooling*
B	*Max Pooling*

Skenario kedua yaitu peneliti akan mencoba untuk melakukan pelatihan model dengan menggunakan jumlah *encoder* dan *decoder* yang bervariatif. Skenario pengujian untuk jumlah *encoder* dan *decoder* dapat dilihat pada tabel 3.5.

Tabel 3.5 Skenario Kedua

Kode Skenario	*Encoder*	*Decoder*	Keterangan
1	1 *Block*	1 *Block*	9 *Hidden layer*
2	2 *Block*	2 *Block*	15 *Hidden layer*
3	3 *Block*	3 *Block*	21 *Hidden layer*
4	4 *Block*	4 *Block*	27 *Hidden layer*

Dalam 1 *block encoder* maupun *decoder* terdapat 3 *hidden layer* didalamnya. Setiap arsitektur di semua skenario terdapat 1 *block bottleneck* yang berisi 2 *hidden layer* dan 1 *hidden layer* terakhir untuk *layer* aktivasi yang terletak di akhir *layer*. Sehingga total *hidden layer* untuk setiap skenario selalu ganjil dengan deskripsi sebagai berikut :

1. Kode skenario 1 dengan 1 *block encoder* (3 *hidden layer*) + 1 *block decoder* (3 *hidden layer*) + 1 *block bottleneck* (2 *hidden layer*) + 1 aktivasi *layer*, sehingga total terdapat 9 *hidden layer*
2. Kode skenario 2 dengan 2 *block encoder* (6 *hidden layer*) + 2 *block decoder* (6 *hidden layer*) + 1 *block bottleneck* (2 *hidden layer*) + 1 aktivasi *layer*, sehingga total terdapat 15 *hidden layer*.
3. Kode skenario 3 dengan 3 *block encoder* (9 *hidden layer*) + 3 *block decoder* (9 *hidden layer*) + 1 *block bottleneck* (2 *hidden layer*) + 1 aktivasi *layer*, sehingga total terdapat 21 *hidden layer*.

4. Kode skenario 4 dengan 4 *block encoder* (12 *hidden layer*) + 4 *block decoder* (12 *hidden layer*) + 1 *block bottleneck* (2 *hidden layer*) + 1 aktivasi *layer*, sehingga total terdapat 27 *hidden layer*

Tabel 3.6 Kombinasi Skenario *Average pooling*

Skenario	Kombinasi	Keterangan
1	1-A	9 *hidden layer* dengan menggunakan *average pooling*
2	2-A	15 *hidden layer* dengan menggunakan *average pooling*
3	3-A	21 *hidden layer* dengan menggunakan *average pooling*
4	4-A	27 *hidden layer* dengan menggunakan *average pooling*

Tabel 3.6 merupakan kombinasi skenario untuk *average pooling*, dimana untuk semua *pooling layer* akan menggunakan *average pool* sedangkan untuk *hidden layer* akan berbeda pada setiap skenarionya. Nantinya hasil dari kombinasi ini akan di analisis dan dibandingkan dengan sesama skenario pada kombinasi skenario *average pooling*.

Tabel 3.7 Kombinasi Skenario *Max Pooling*

Skenario	Kombinasi	Keterangan
5	1-B	9 *hidden layer* dengan menggunakan *max pooling*
6	2-B	15 *hidden layer* dengan menggunakan *max pooling*
7	3-B	21 *hidden layer* dengan menggunakan *max pooling*
8	4-B	27 *hidden layer* dengan menggunakan *max pooling*

Tabel 3.7 merupakan kombinasi skenario untuk *max pooling*, dimana untuk semua *pooling layer* akan menggunakan *max pool* sedangkan untuk *hidden layer* akan berbeda pada setiap skenarionya. Nantinya hasil dari kombinasi ini akan di

analisis dan dibandingkan dengan sesama skenario pada kombinasi skenario *max pooling.*

BAB IV

HASIL DAN PEMBAHASAN

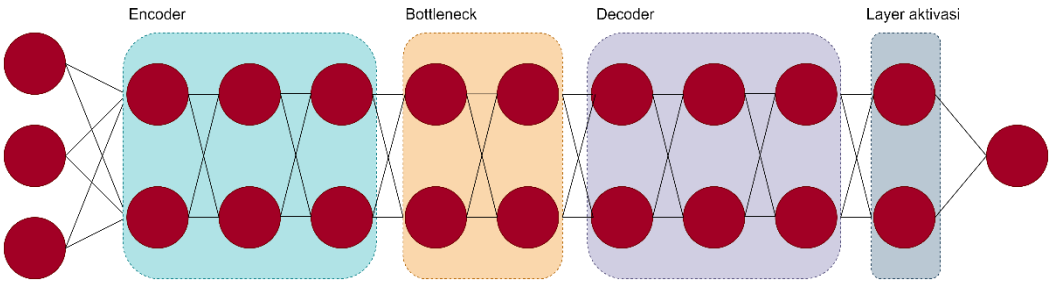
4.1 Langkah Pengujian

Langkah – langkah pengujian akan dilakukan berdasarkan skenario pengujian yang telah diuraikan di bab sebelumnya. Berikut adalah langkah pengujian yang sudah dilakukan :

a. Proses *Training*, pada tahap ini dilakukan training data yang telah melalui preprocessing yang sudah dijelaskan pada 3.2.2. Data akan di training dengan menggunakan model *convolutional neural network*. Setelah model dilatih, maka akan dilakukan proses *testing*.
b. Proses *Testing*, pada tahap ini model yang sudah dilatih akan mendapatkan bobot yang paling optimal. Maka langkah selanjutnya adalah melakukan predict terhadap unseen data untuk dilihat seberapa bagus nilai akurasi dan nilai MioU.
c. Analisis hasil pengujian, pada tahap ini akan ditampilkan hasil dari setiap skenario pengujian dalam bentuk tabel. Hasil pengujian yang akan ditampilkan adalah akurasi, *loss*, dan nilai MioU. Pada proses ini akan terlihat mana skenario pengujian yang mendapatkan hasil terbaik.

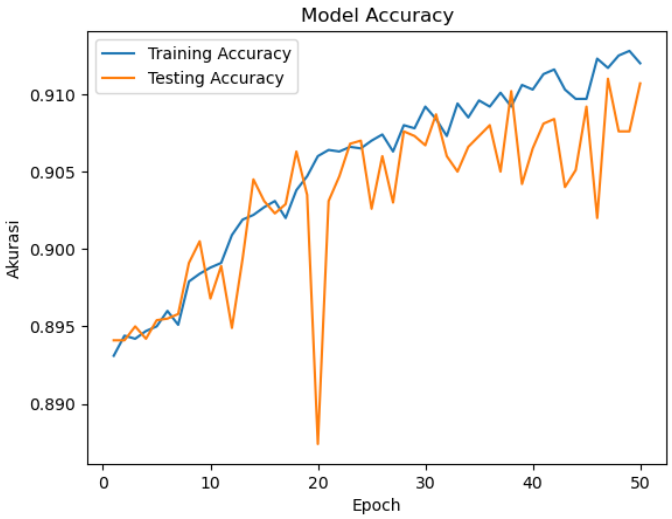
4.2 Hasil Pengujian

A. Skenario pertama (1-A)



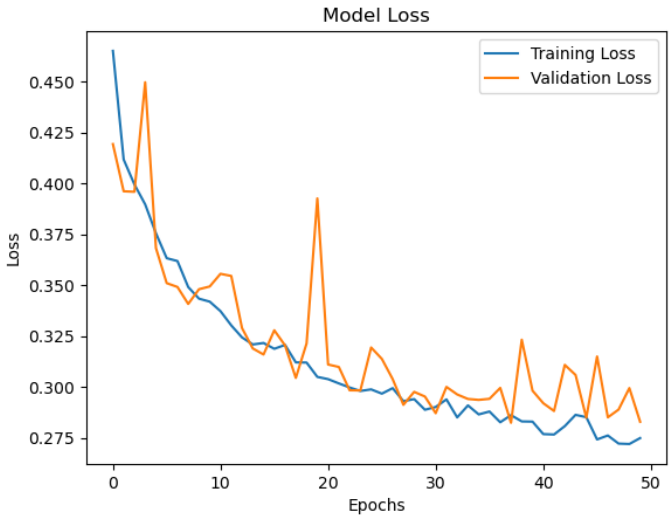
Gambar 4.1 Arsitektur 9 hidden layer

Pada skenario pertama, model *convolutional neural network* menggunakan masing – masing 1 blok *encoder* dan *decoder* sehingga terdapat total 9 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *average pool* pada proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9120 ditampilkan dalam gambar 4.2 :



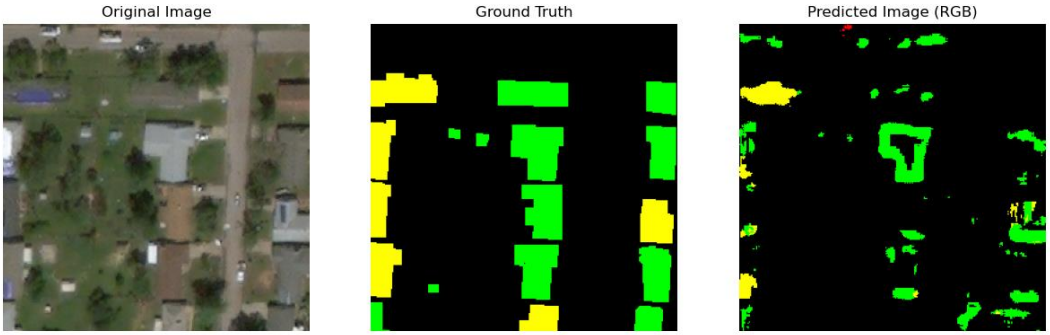
Gambar 4.2 Akurasi skenario 1-A

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.2749 dan ditampilkan dalam gambar 4.3 :

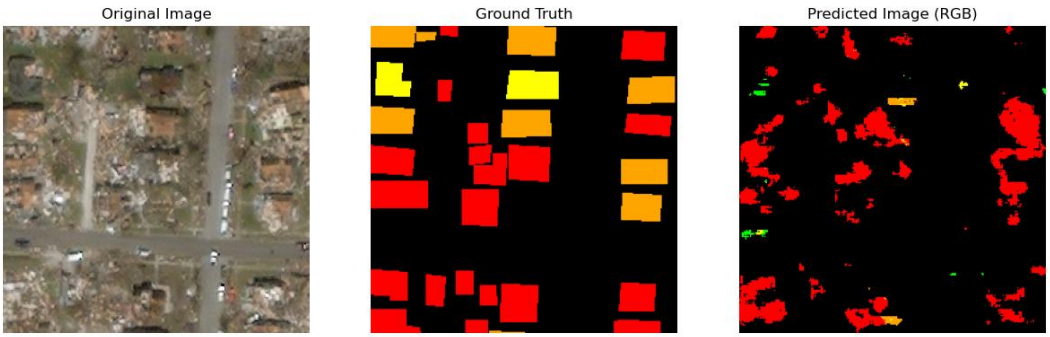


Gambar 4.3 Loss skenario 1-A

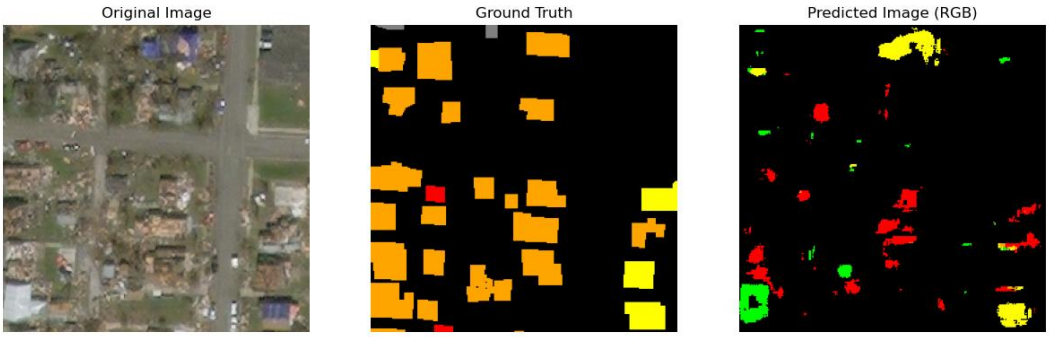
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus nilai MioU :



Gambar 4.4 Prediksi pertama skenario 1-A



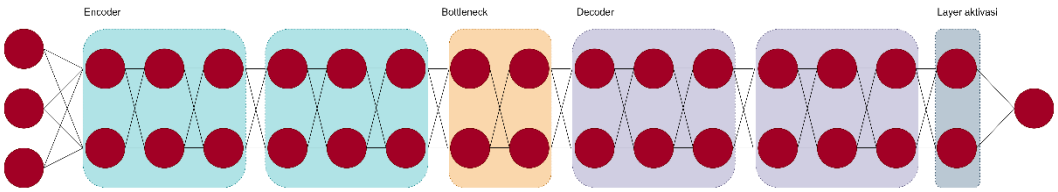
Gambar 4.5 prediksi kedua skenario 1-A



Gambar 4.6 Prediksi ketiga skenario 1-A

Bisa disimpulkan pada skenario 1-A model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9120 dan nilai loss sebesar 0.2749. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.25.

B. Skenario Kedua (2-A)



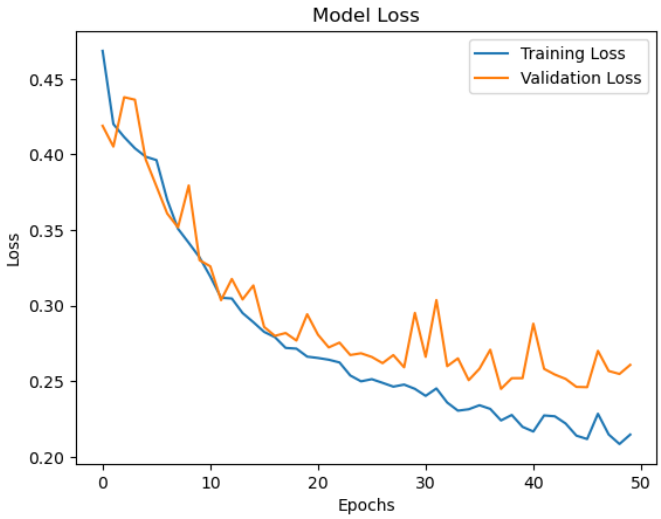
Gambar 4.7Arsitektur 15 hidden layer

Pada skenario kedua, model *convolutional neural network* menggunakan masing – masing 2 blok *encoder* dan *decoder* sehingga terdapat total 15 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *average pool* pada

proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9281 ditampilkan dalam gambar 4.8 :

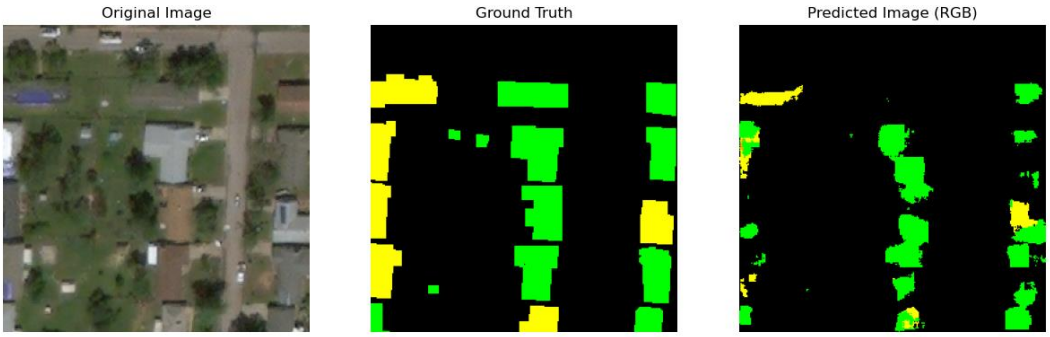
Gambar 4.8 Akurasi skenario 2-A

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.2147 dan ditampilkan dalam gambar 4.9 :

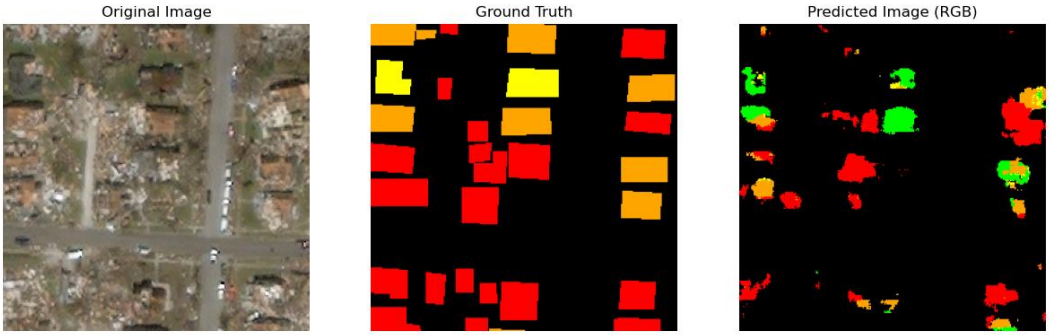


Gambar 4.9 Loss skenario 2-A

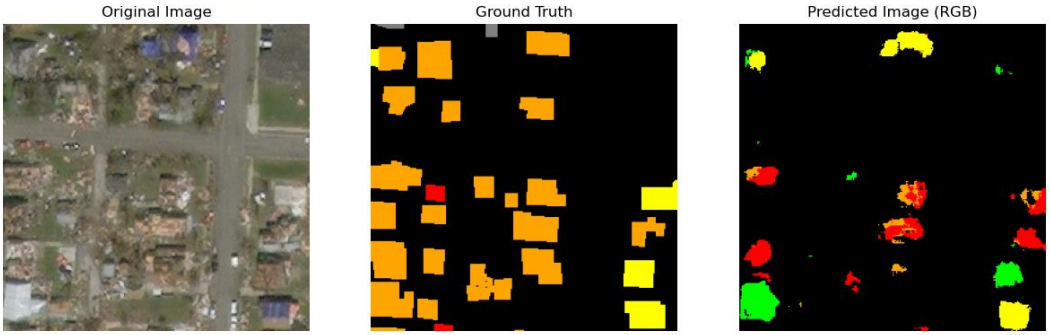
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan MIoU :



Gambar 4.10 Prediksi pertama skenario 2-A



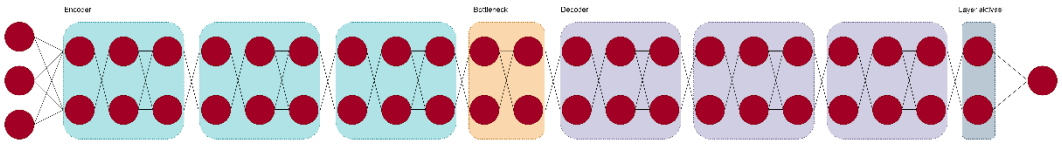
Gambar 4.11 Prediksi kedua skenario 2-A



Gambar 4.12 Prediksi ketiga skenario 2-A

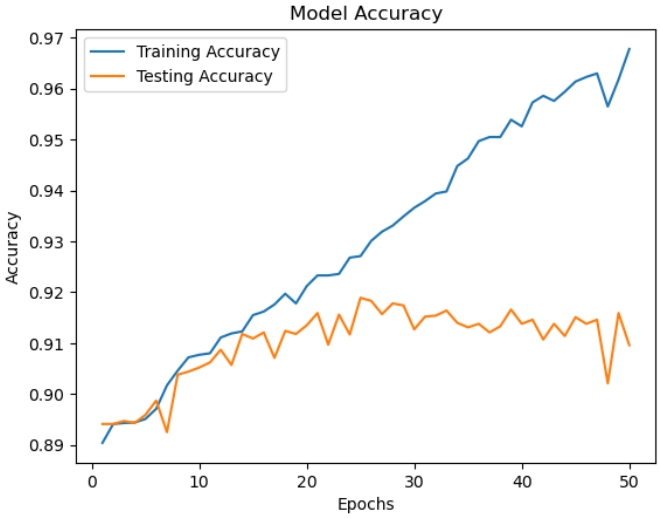
Bisa disimpulkan pada skenario 2-A model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9281 dan nilai *loss* sebesar 0.2147. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.28.

C. Skenario Ketiga (3-A)



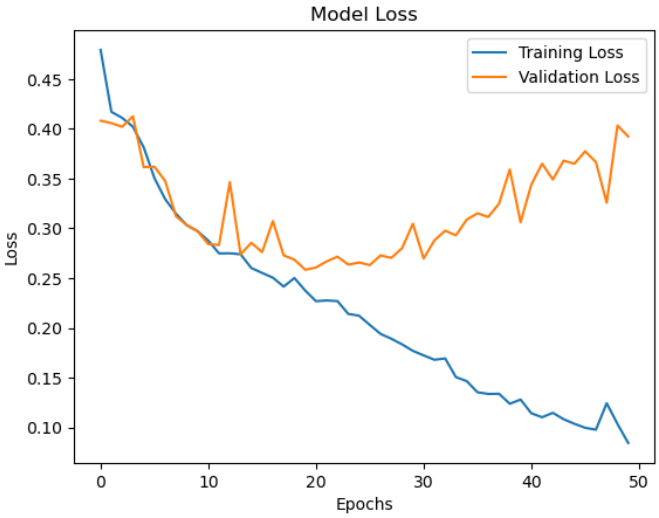
Gambar 4.13 Arsitektur 21 hidden layer

Pada skenario ketiga, model *convolutional neural network* menggunakan masing – masing 3 blok *encoder* dan *decoder* sehingga terdapat total 21 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *average pool* pada proses *pooling.* Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9678 ditampilkan dalam gambar 4.14 :



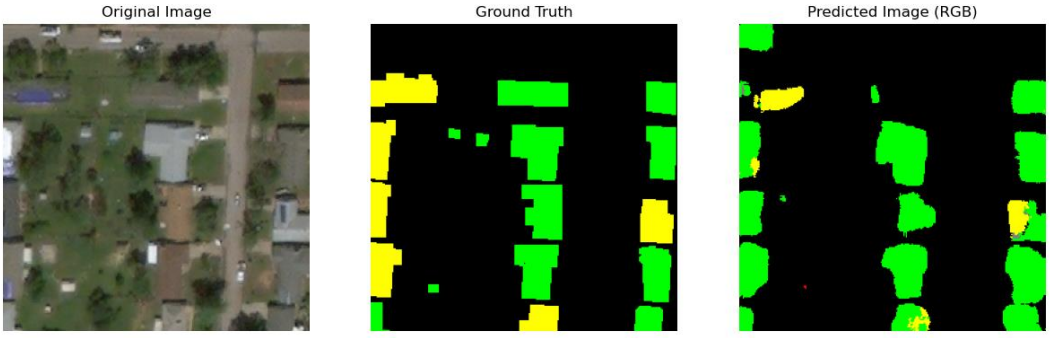
Gambar 4.14 Akurasi skenario 3-A

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.0844 dan ditampilkan dalam gambar 4.15 :

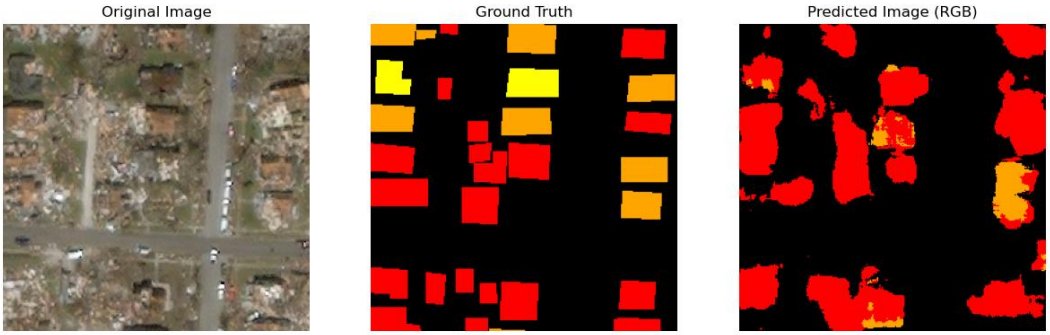


Gambar 4.15 Loss skenario 3-A

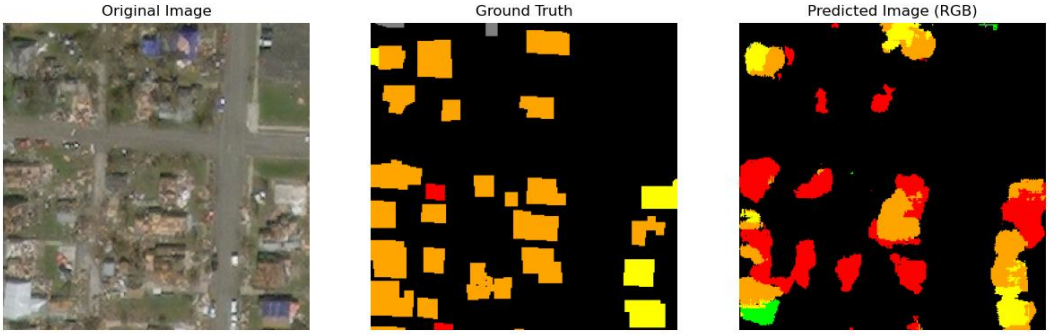
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan MIoU :



Gambar 4.16 Prediksi pertama skenario 3-A



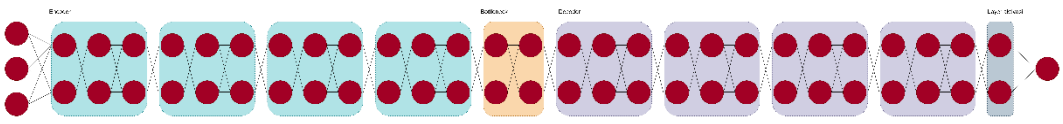
Gambar 4.17 Prediksi kedua sekeario 3-A



Gambar 4.18 Prediksi ketiga skenario 3-A

Bisa disimpulkan pada skenario 3-A model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9678 dan nilai *loss* sebesar 0.0844. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.25.

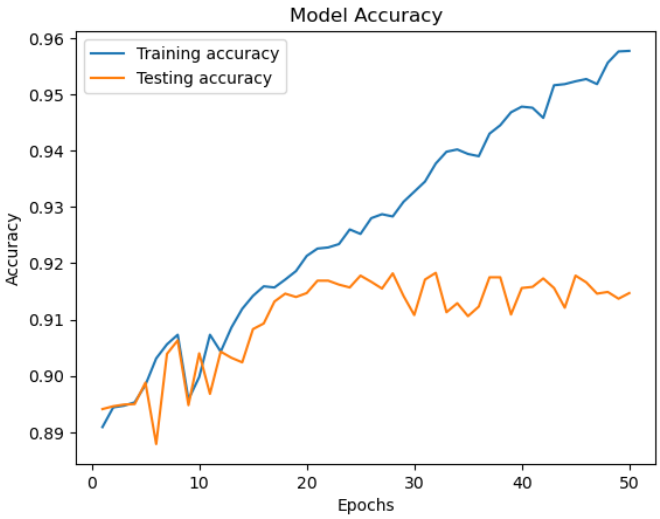
D. Skenario Keempat (4-A)



Gambar 4.19 Arsitektur 27 hidden layer

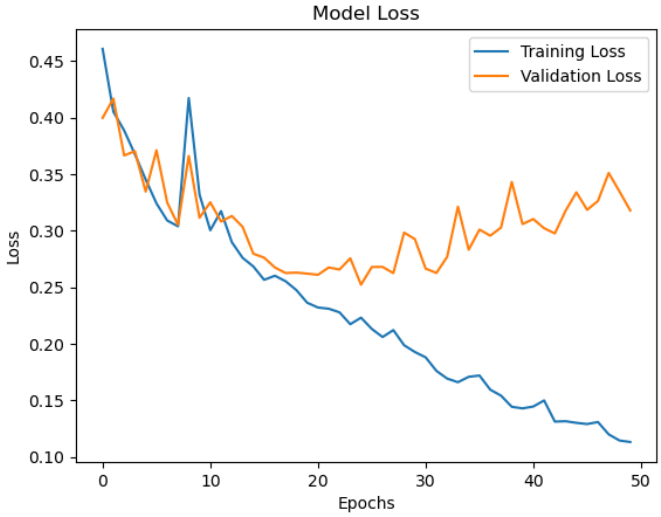
Pada skenario keempat, model *convolutional neural network* menggunakan masing – masing 4 blok *encoder* dan *decoder* sehingga terdapat total 27 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *average*

pool pada proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9577 ditampilkan dalam gambar 4.20 :



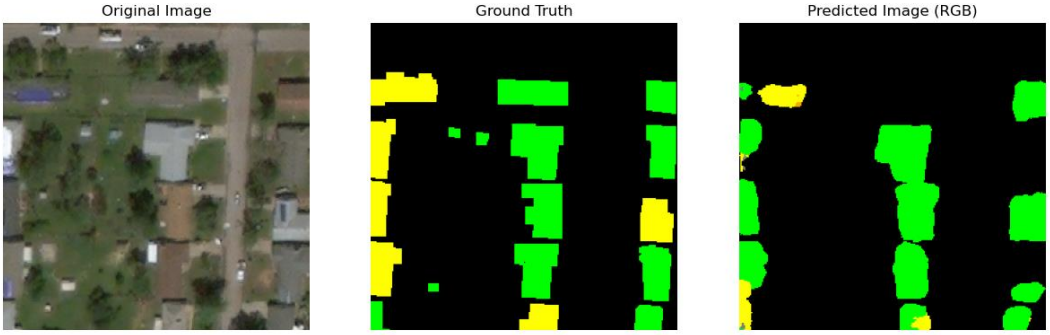
Gambar 4.20 Akurasi skenario 4-A

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.1133 dan ditampilkan dalam gambar 4.21 :

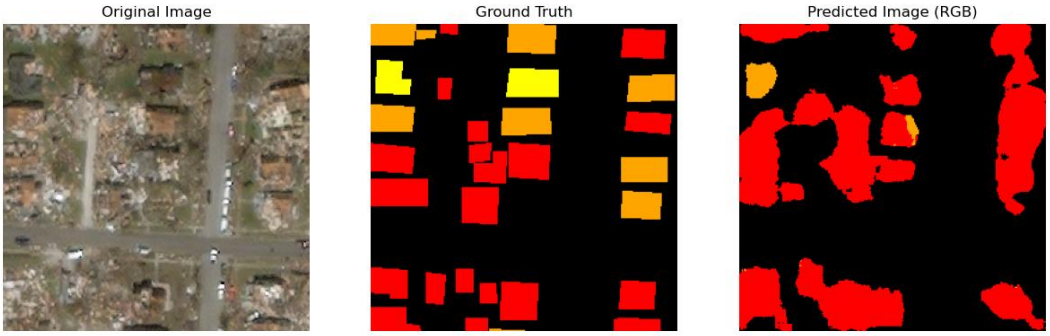


Gambar 4.21 Loss skenario 4-A

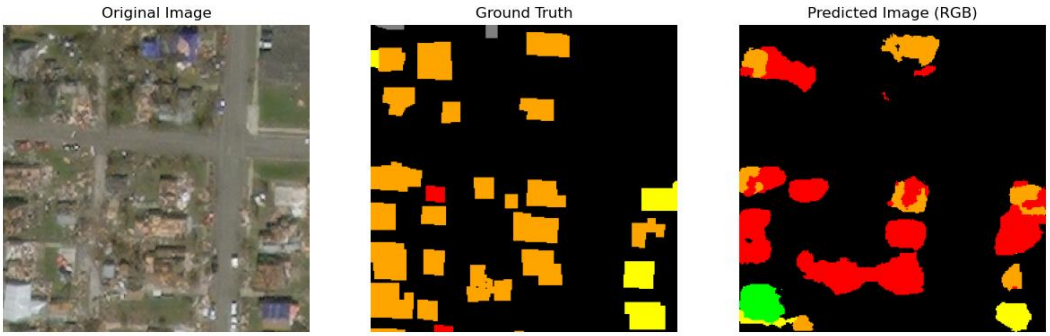
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan nilai MIoU :



Gambar 4.22 Prediksi pertama skenario 4-A



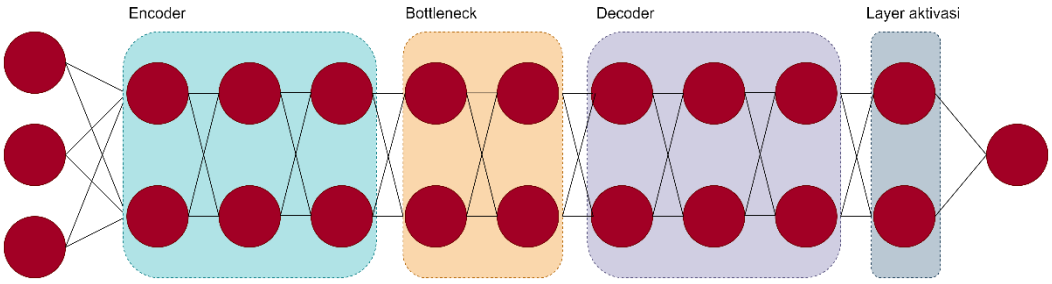
Gambar 4.23 Prediksi kedua skenario 4-A



Gambar 4.24 Prediksi ketiga skenario 4-A

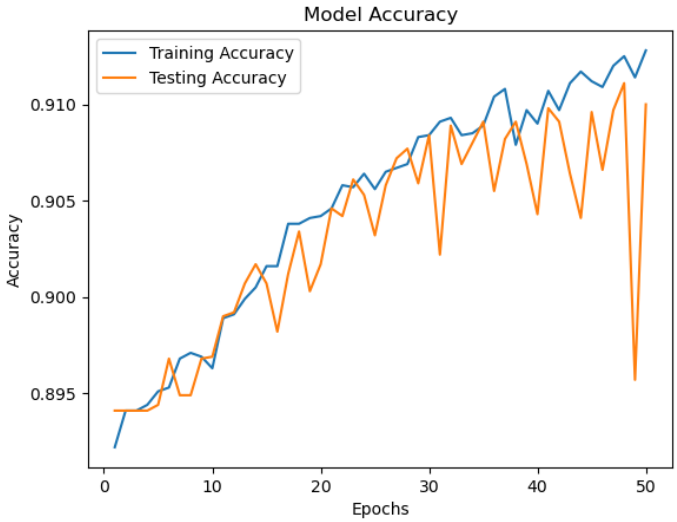
Bisa disimpulkan pada skenario 4-A model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9577 dan nilai *loss* sebesar 0.1133. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.31.

E. Skenario Kelima (1-B)



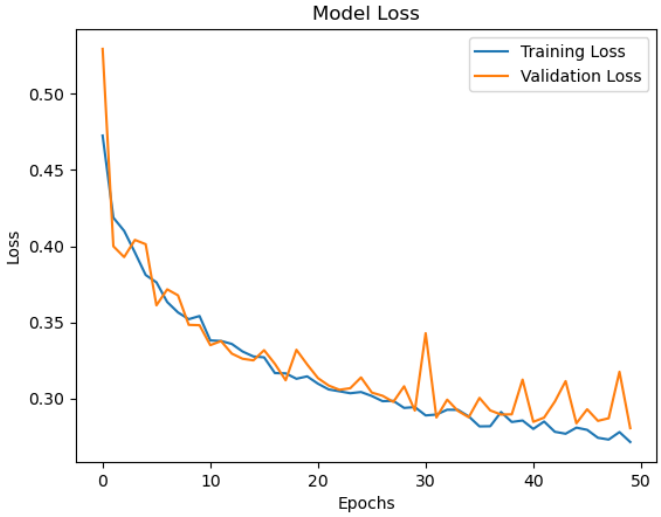
Gambar 4.25 Arsitektur 9 hidden layer

Pada skenario kelima, model *convolutional neural network* menggunakan masing – masing 1 blok *encoder* dan *decoder* sehingga terdapat total 8 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *max pool* pada proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9128 ditampilkan dalam gambar 4.26 :



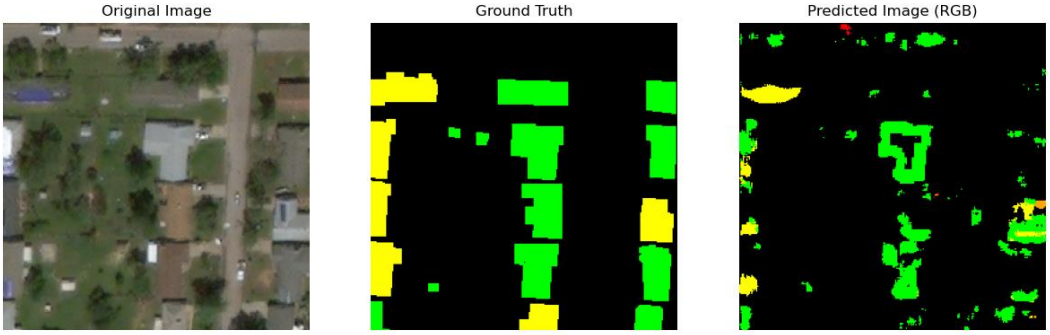
Gambar 4.26 Akurasi skenario 1-B

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.2715 dan ditampilkan dalam gambar 4.27 :

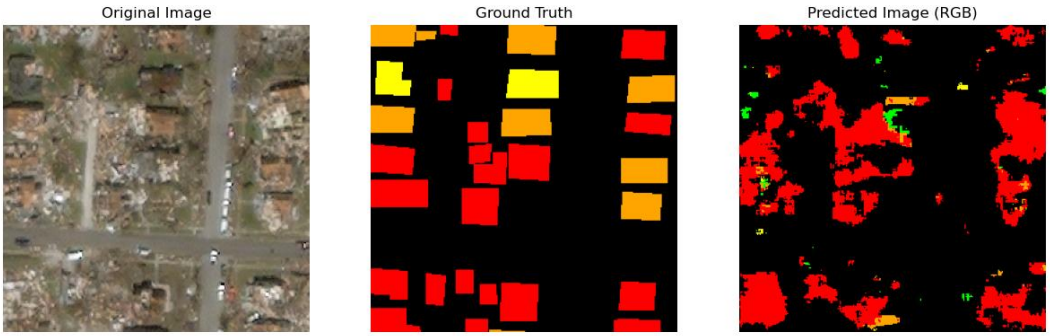


Gambar 4.27 Loss skenario 1-B

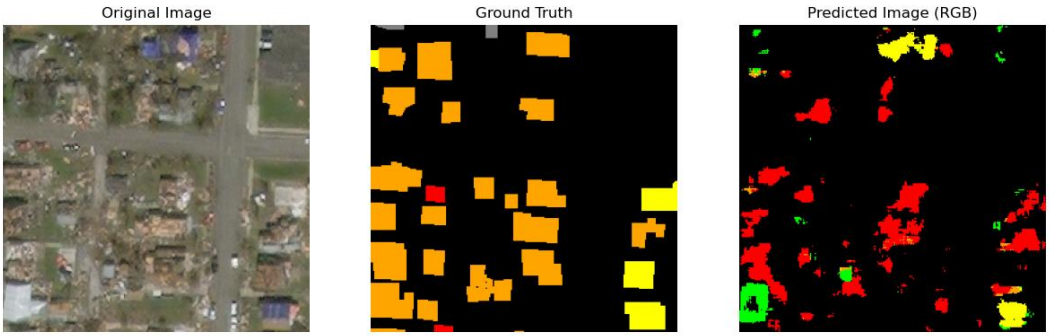
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan nilai MIoU :



Gambar 4.28 Prediksi pertama skenario 1-B



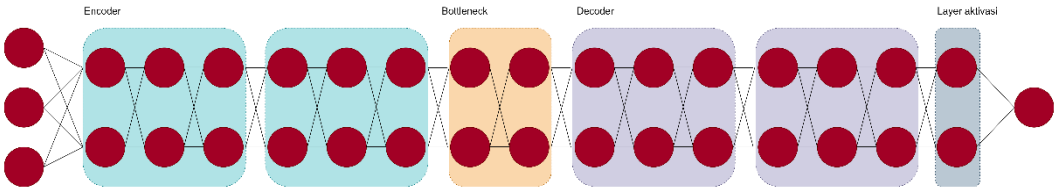
Gambar 4.29 Prediksi kedua skenario 1-B



Gambar 4.30 Prediksi ketiga skenario 1-B

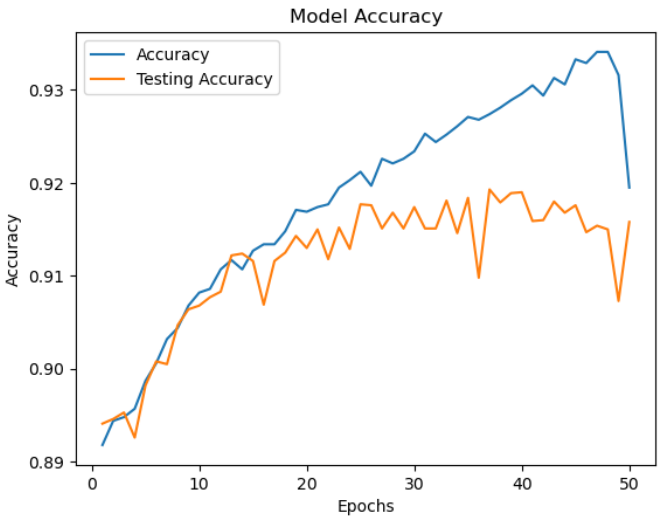
Bisa disimpulkan pada skenario 1-B model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9128 dan nilai loss sebesar 0.2715. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.24.

F. Skenario Keenam (2-B)



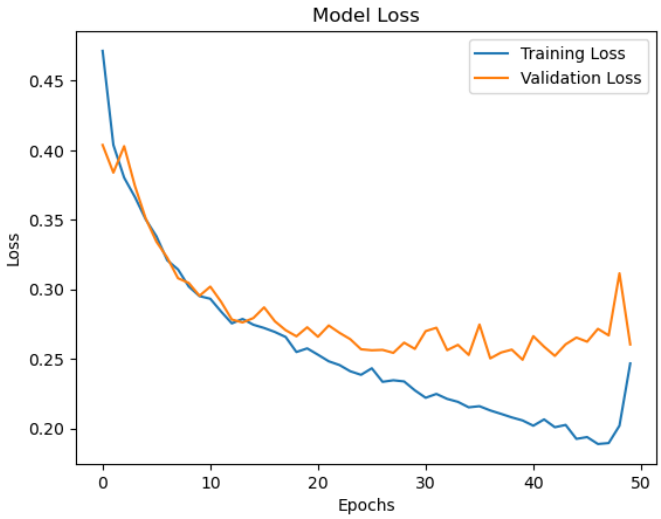
Gambar 4.31 Arsitektur 15 hidden layer

Pada skenario keenam, model *convolutional neural network* menggunakan masing – masing 2 blok *encoder* dan *decoder* sehingga terdapat total 15 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *max pool* pada proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9195 ditampilkan dalam gambar 4.32 :



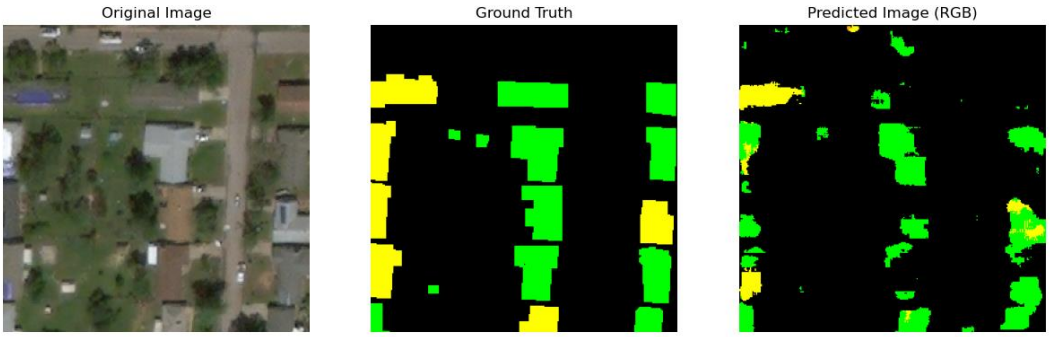
Gambar 4.32 Akurasi skenario 2-B

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.2468 dan ditampilkan dalam gambar 4.33 :

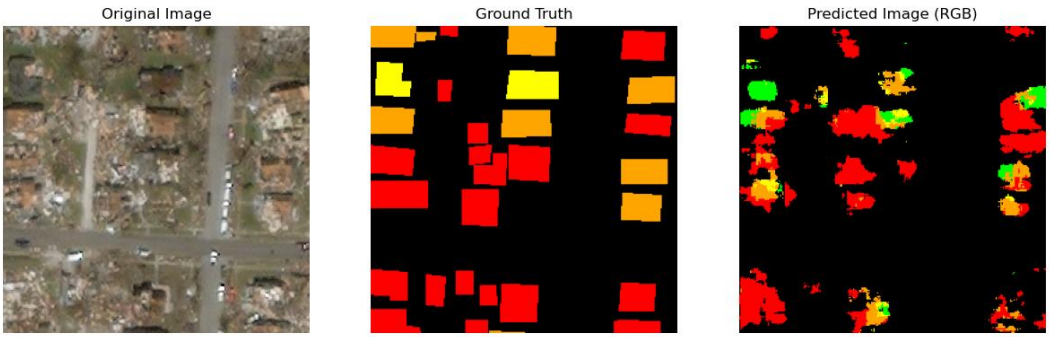


Gambar 4.33 Loss skenario 2-B

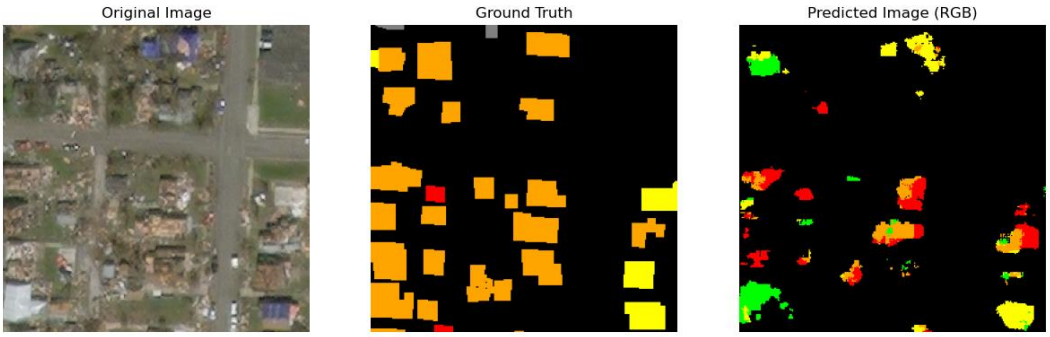
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan MIoU :



Gambar 4.34 Prediksi pertama skenario 2-B



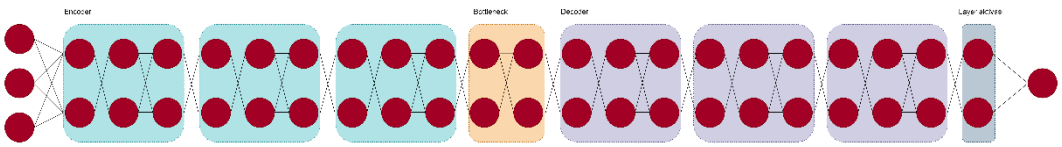
Gambar 4.35 Prediksi kedua skenario 2-B



Gambar 4.36 Prediksi ketiga skenario 2-B

Bisa disimpulkan pada skenario 2-B model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9195 dan nilai *loss* sebesar 0.2468. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.28.

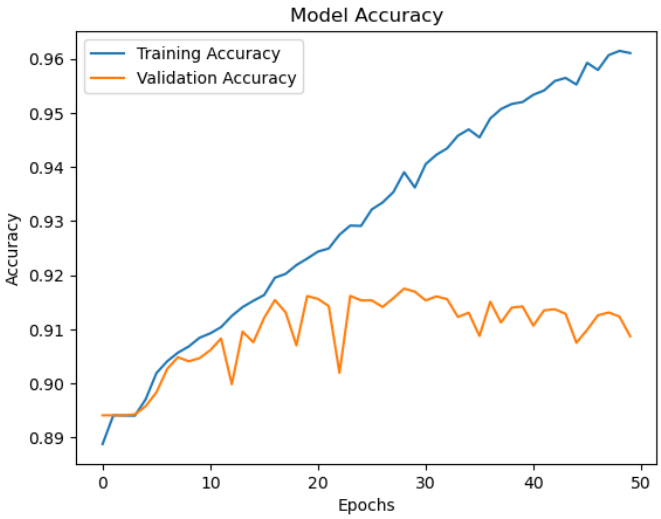
G. Skenario Ketujuh (3-B)



Gambar 4.37 Arsitektur 21 hidden layer

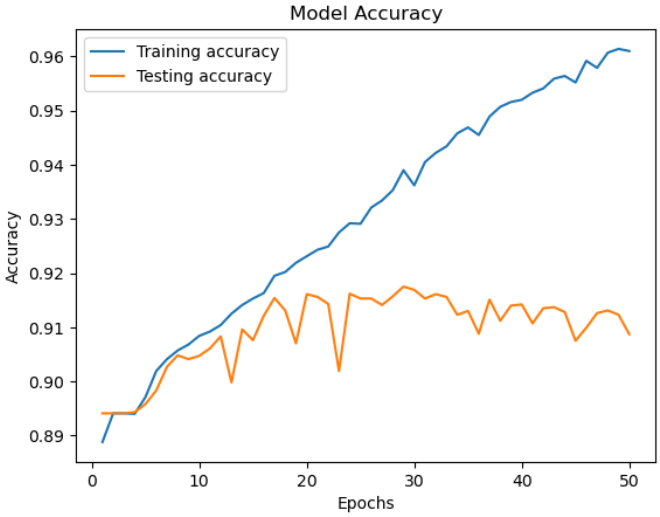
Pada skenario ketujuh, model *convolutional neural network* menggunakan masing – masing 3 blok *encoder* dan *decoder* sehingga terdapat total 21 *hidden layer* pada arsitektur ini. Pada skenario ini serta menggunakan *max pool* pada

proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9610 ditampilkan dalam gambar 4.38 :



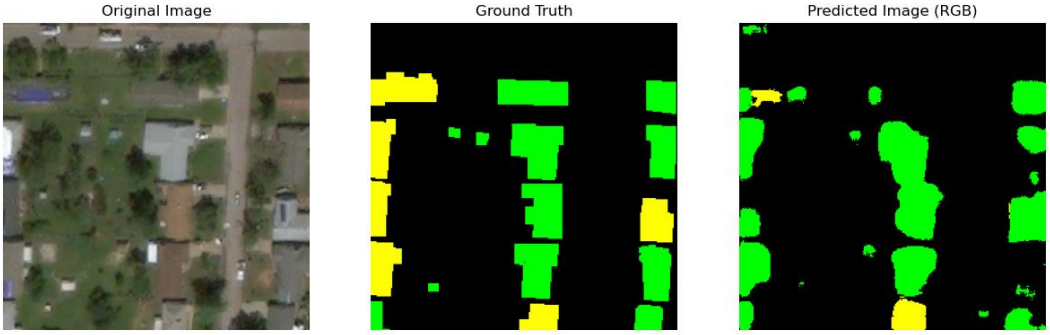
Gambar 4.38 Akurasi skenario 3-B

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.1057 dan ditampilkan dalam gambar 4.39 :

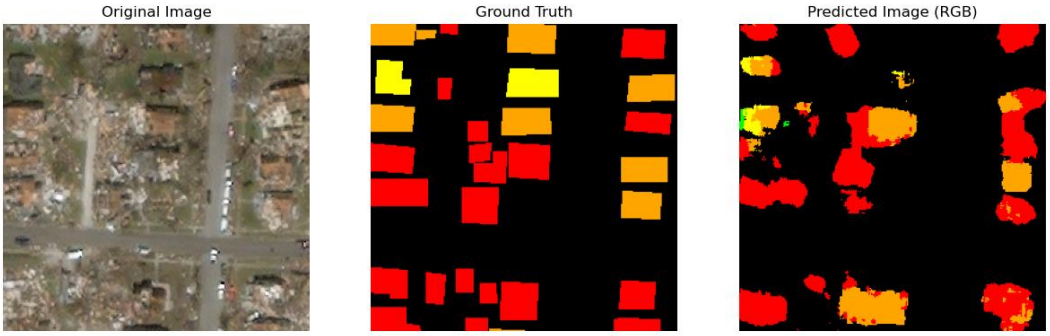


Gambar 4.39 Loss skenario 3-B

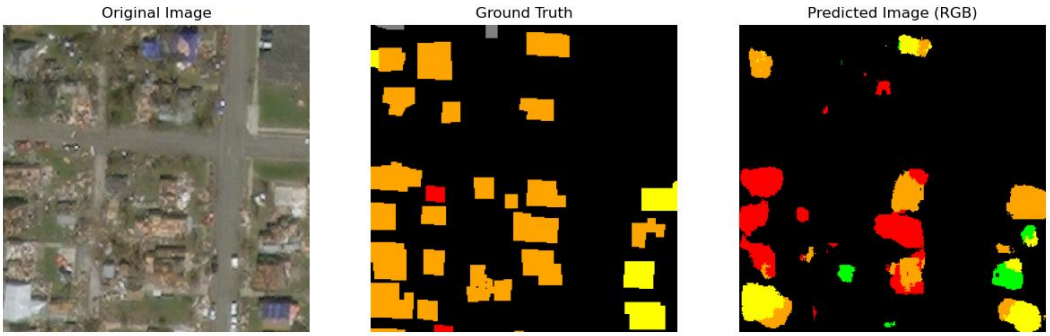
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan MIoU :



Gambar 4.40 Prediksi pertama skenario 3-B



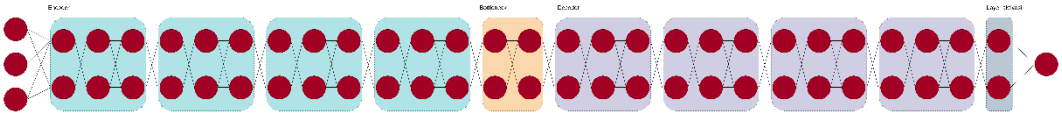
Gambar 4.41 Prediksi kedua skenario 3-B



Gambar 4.42 Prediksi ketiga skenario 3-B

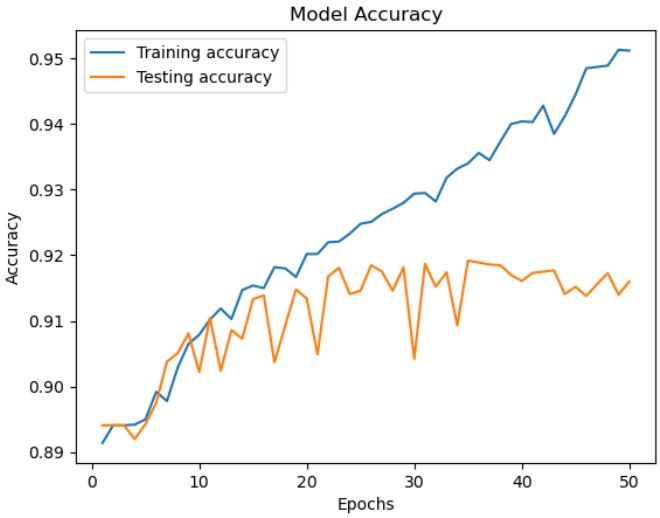
Bisa disimpulkan pada skenario 3-B model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9610 dan nilai *loss* sebesar 0.1057. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.29.

H. Skenario Kedelapan (4-B)



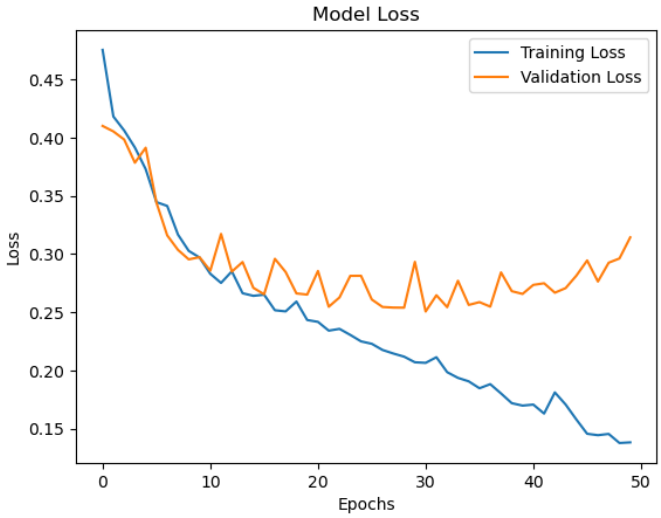
Gambar 4.43 Arsitektur 27 hidden layer

Pada skenario kedelapan, model *convolutional neural network* menggunakan masing – masing 4 blok *encoder* dan *decoder* sehingga terdapat total 27 *hidden layer* pada arsitektur ini. Pada skenario ini menggunakan *max pool* pada proses *pooling*. Akurasi model setelah *training* dengan data sebanyak 1400 data dan dengan data testing sebanyak 600 data menghasilkan nilai sebesar 0.9512 ditampilkan dalam gambar 4.36 :



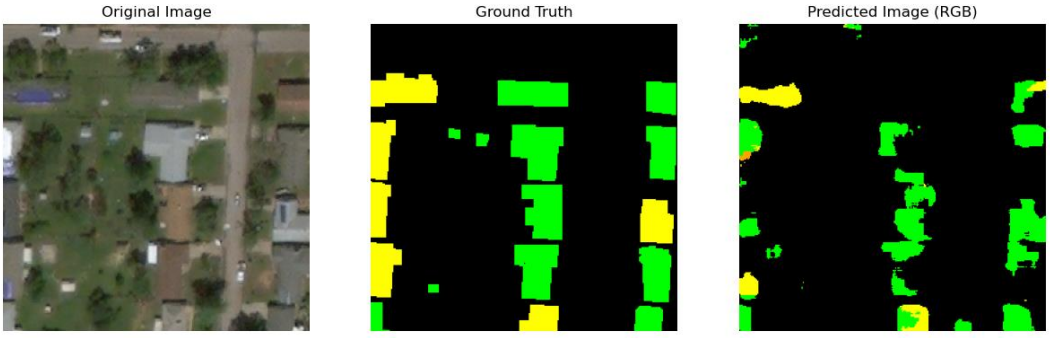
Gambar 4.44 Akurasi skenario 4-B

Berikutnya yaitu nilai *loss* yang didapatkan adalah bernilai 0.1381 dan ditampilkan dalam gambar 4.37 :

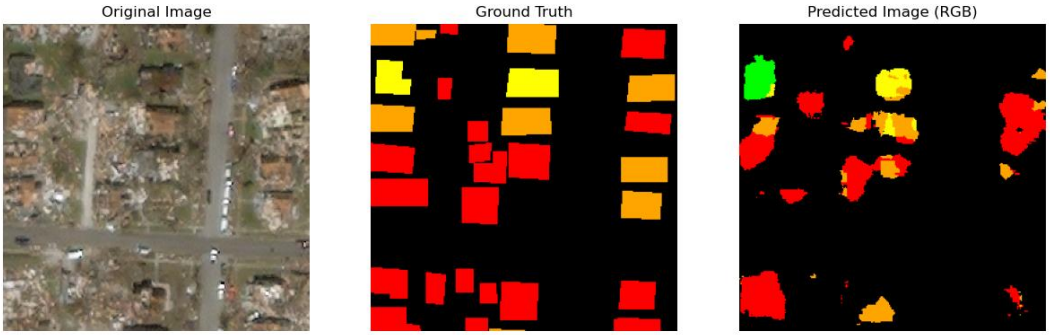


Gambar 4.45 Loss Skenario 4-B

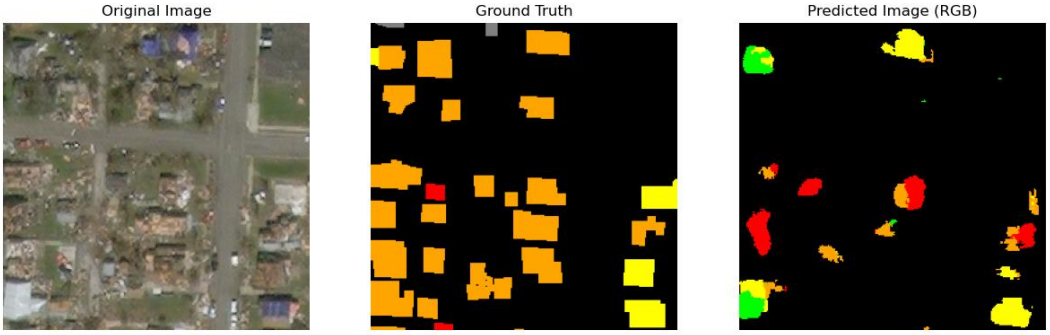
Langkah selanjutnya adalah proses *predict* dengan *unseen data* sebanyak 320 data untuk mendapatkan nilai MIoU . Hasil *predict* sekaligus untuk mendapatkan MIoU :



Gambar 4.46 Prediksi pertama skenario 4-B



Gambar 4.47 Prediksi kedua skenario 4-B



Gambar 4.48 Prediksi ketiga skenario 4-B

Bisa disimpulkan pada skenario 1-A model *convolutional neural network* berhasil mendapatkan nilai akurasi sebesar 0.9512 dan nilai *loss* sebesar 0.1381. Pada proses *predict*, model pada skenario ini mendapatkan nilai MioU sebesar 0.28.

Jika disusun dalam bentuk tabel, berikut adalah hasil akurasi, *loss*, dan MioU dari masing – masing skenario pengujian pada tabel 4.1 dan tabel 4.2 :

Tabel 4.1 Hasil Pengujian *Average pooling*

Nomor	Skenario	Akurasi	Loss	MioU
1	1-A (9 hidden layer)	0.9120	0.2749	0.25
2	2-A (15 hidden layer)	0.9281	0.2147	0.28
3	3-A (21 hidden layer)	**0.9678**	0.0844	0.25
4	4-A (27 hidden layer)	0.9577	0.1133	**0.31**

Tabel 4.2 Hasil Pengujian *Max Pooling*

Nomor	Skenario	Akurasi	Loss	MioU
5	1-B (9 hidden layer)	0.9128	0.2715	0.24
6	2-B (15 hidden layer)	0.9195	0.2468	0.28
7	3-B (21 hidden layer)	**0.9610**	0.1057	**0.29**
8	4-B (27 hidden layer)	0.9512	0.1381	0.28

Akurasi terbesar yang diperoleh pada skenario pengujian *average pooling* didapatkan pada skenario pengujian 3-A dengan 21 *hidden layer*. Untuk MioU terbaik didapatkan pada skenario pengujian 4-A dengan 27 *hidden layer*. Sedangkan pada skenario *max pooling*, akurasi terbesar sekaligus nilai MioU terbaik didapatkan pada skenario 3-B dengan 21 *hidden layer*.

4.3 Pembahasan

Berdasarkan tabel pengujian *average pooling* pada tabel 4.1, masing – masing skenario telah berhasil mendapatkan nilai akurasi dan MioU. Meskipun di training dengan hyperparameter dan data yang sama, nampaknya arsitektur yang berbeda juga sangat mempengaruhi hasil dari kinerja model baik selama proses training maupun proses *predict*. Berdasarkan penelitian ini juga dapat dilihat bahwa banyaknya *hidden layer* tidak selamanya memberikan hasil yang bagus juga untuk semua nilai. Terlihat pada hasil pengujian di tabel 4.1, akurasi tertinggi didapatkan pada skenario 3-A dengan 21 hidden layer dengan akurasi sebesar 0.9678, sedangkan MioU terbaik didapatkan oleh arsitektur yang diajukan penulis pada sub bab 3.2.3 berhasil memperoleh 0.31. Melalui hasil pengujian ini dapat disimpulkan bahwa banyaknya hidden layer untuk *average pooling* hanya berpengaruh pada MioU, semakin banyak hidden layer maka semakin bagus nilai MioU. Berbanding dengan akurasi, banyaknya hidden layer tidak mempengaruhi

nilai akurasi, karena menurut pengujian akurasi dengan 21 hidden layer mendapatkan nilai yang lebih tinggi yaitu 0.9678 ketimbang skenario dengan 27 hidden layer yang hanya mendapatkan akurasi 0.9577.

Pada tabel pengujian 4.2 untuk skenario pengujian *max pooling*, akurasi terbesar didapatkan oleh skenario 3-B dengan 21 hidden layer yaitu mendapatkan akurasi sebesar 0.9610. Nilai MioU terbesar pada skenario *max pooling* juga didapatkan pada skenario yang sama yaitu pada 3-B dengan nilai MioU sebesar 0.29. Berdasarkan hasil pengujian ini, dapat disimpulkan bahwa pada skenario *max pooling* jumlah hidden layer tidak berpengaruh pada nilai akurasi dan MioU. Semakin banyak hidden layer tidak mempengaruhi nilai akurasi dan MioU, dapat dilihat bahwa nilai akurasi dan MioU untuk skenario 3-B dengan 21 hidden layer mendapatkan nilai yang lebih tinggi ketimbang skenario 4-B dengan hanya mendapatkan nilai akurasi 0.9512 dan nilai MioU 0.28. Pengaruh jumlah hidden layer dalam hal MioU untuk skenario *max pooling* berbanding terbalik dengan skenario *average pooling*. Namun, pengaruh hidden layer terhadap akurasi pada kedua skenario ini mendapatkan kesimpulan yang sama.

Pada proses *predict*, masing – masing model pada skenario pengujian menampilkan hasil prediksi segmentasinya berupa gambar segmentasi yang dapat dilihat pada sub bab 4.2. Meskipun nilai akurasi tinggi, nampaknya model dengan *hidden layer* yang lebih sedikit masih kesulitan dalam memprediksi bangunan dan tingkat kerusakannya. Pada skenario *average pooling* dapat dilihat hasil prediksi untuk skenario 1-A dan 2-A masih kesulitan dalam memprediksi bentuk dari bangunan. Namun pada skenario 3-A dan 4-A, model sudah mulai mampu

memprediksi bentuk bangunan yang lebih baik ketimbang model skenario 1-A dan 2-A.

Demikian pula, pada skenario *max pooling* 1-B dan 2-B, model masih menghadapi kesulitan dalam memprediksi bentuk bangunan. Namun, pada skenario 3-B dan 4-B, model mulai menunjukkan kemampuan yang lebih baik dalam memprediksi bentuk bangunan. Mengingat kedua skenario tersebut menunjukkan pola yang sama, dapat disimpulkan bahwa semakin banyak lapisan tersembunyi yang digunakan, semakin mampu model dalam memprediksi bentuk bangunan.

Terlihat pada beberapa skenario pengujian baik itu pada *average pooling* maupun *max pooling*, nampaknya model masih sedikit kesulitan ketika memprediksi antara kelas no-damage dengan *minor-damage* serta kelas *major-damage* dengan *destroyed*. Pada gambar 4.16 dan 4.12 untuk *average pooling* serta gambar 4.40 untuk *max pooling*, terlihat bahwa rumah yang seharusnya *minor-damage* diidentifiaksi sebagai *no-damage*. Hal yang sama juga terjadi pada gambar 4.17 untuk *average pooling* serta 4.41 untuk *max pooling*, namun kali ini rumah yang seharusnya *major-damage* justru diidentifikasi sebagai *destroyed.*

Terlihat bahwa model beberapa kali memprediksi kekeliruan kelas yang hampir mirip seperti halnya *minor-damage* dan *no-damage* atau *major-damage* dan *destroyed*. Jika kita melihat datanya sebagai manusia memang terlihat baik bentuk dan tekstur dari *minor-damage* dan *no-damage* hampir sama, demikian juga dengan *major-damage* dan *destroyed*. Solusi untuk menyelesaikan permasalahan ini adalah dengan memperbanyak data latih sehingga model

mendapatkan lebih banyak informasi agar dapat memprediksi setiap kelasnya. Selain itu juga, model juga nampaknya masih kesusahan dalam memprediksi bentuk bangunan terutama pada kelas *major-damage* dan *destroyed*.

Berikutnya pada grafik akurasi dan *loss*, pada skenario *average pooling* 3-A dan 4-A terlihat bahwa terdapat jarak yang cukup signifikan antara grafik akurasi dan akurasi *testing* beserta grafik *loss* dan *val loss*. Hal ini berbeda dengan skenario 1-A dan 2-A yang memiliki sedikit jarak antara akurasi dan akurasi *testing* serta *loss* dengan *val loss*.

Pada skenario *max pooling* juga menghasilkan pola grafik yang sama, yaitu pada skenario 3-B dan 4-B memiliki jarak yang cukup signifikan antara grafik akurasi dan akurasi *testing* beserta grafik *loss* dan *val loss*. Sedangkan untuk 1-B dan 2-B, akurasi dan akurasi testing serta loss dan val loss memiliki sedikit jarak. Melihat pola yang sama ini maka dapat disimpulkan bahwa baik skenario *average pooling* maupun *max pooling*, kompleksitas model sangat berpengaruh terhadap data yang diberikan. Hal ini disebabkan karena terlalu kompleksnya model arsitektur yang tidak sebanding dengan kompleksitas data yang diberikan. Kejadian ini disebut dengan *overfitting*, yaitu situasi di mana model sangat sesuai dengan data pelatihan sehingga kehilangan kemampuan untuk melakukan generalisasi terhadap data baru atau data uji. Fenomena overfitting sering terjadi ketika arsitektur model sangat kompleks, namun tidak diimbangi dengan dataset yang kompleks dan cukup besar. Pada penelitian yang dilakukan oleh W. A. Khan *et al*. (2020) mengatakan bahwa Meningkatkan jumlah lapisan, neuron, atau koneksi dapat menyebabkan overfitting, terlepas dari kompleksitas

dataset. Hal ini disebabkan oleh kecenderungan model untuk menyesuaikan diri dengan noise dalam data daripada pola yang mendasarinya, yang berakibat pada penurunan kinerja saat diuji dengan data yang belum pernah dilihat sebelumnya. Mengingat kedua skenario ini menggunakan data yang sama, maka wajar saja jika memang kedua skenario menghasilkan pola yang sama. Untuk mengatasi hal ini yaitu dengan menambahkan data latih beserta meningkatkan jumlah data testing dalam proses *training*.

Dalam konteks deteksi kerusakan, terdapat sebuah narasi historis dalam Al-Qur'an yang mengisahkan perjalanan Nabi Musa AS dan Nabi Khidir AS, sebagaimana tercatat dalam surat Al-Kahfi. Salah satu peristiwa yang diuraikan adalah ketika Nabi Khidir AS memperbaiki sebuah bangunan yang hampir roboh di suatu daerah. Peristiwa ini tercantum dalam surat Al-Kahfi, ayat 77 :

فَٱنطَلَقَا حَتَّىٰٓ إِذَآ أَتَيَآ أَهْلَ قَرْيَةٍ ٱسْتَطْعَمَآ أَهْلَهَا فَأَبَوْا۟ أَن يُضَيِّفُوهُمَا فَوَجَدَا فِيهَا جِدَارًا يُرِيدُ أَن يَنقَضَّ فَأَقَامَهُۥ ۖ قَالَ لَوْ شِئْتَ لَتَّخَذْتَ عَلَيْهِ أَجْرًا

“Maka keduanya berjalan; hingga tatkala keduanya sampai kepada penduduk suatu negeri, mereka minta dijamu kepada penduduk negeri itu, tetapi penduduk negeri itu tidak mau menjamu mereka, kemudian keduanya mendapatkan dalam negeri itu dinding rumah yang hampir roboh, maka Khidhr menegakkan dinding itu. Musa berkata: "Jikalau kamu mau, niscaya kamu mengambil upah untuk itu".” (QS. Al – Kahfi: 77)

Ayat tersebut memiliki tafsir dari Al-Mukhtasar, yang berbunyi: “Maka keduanya melanjutkan perjalanan, hingga ketika keduanya sampai pada suatu negeri, mereka berdua meminta dari penduduk negeri tersebut untuk dijamu, tetapi mereka enggan menjamu mereka berdua. Kemudian di negeri tersebut, keduanya mendapatkan satu dinding rumah yang miring dan hampir jatuh dan

roboh, maka Khaḍir segera memperbaikinya hingga berdiri tegak. Melihat hal ini, Musa berkata kepada Khidir, ‘Jika engkau mau niscaya engkau dapat meminta imbalan dari pembetulan dinding ini karena kita memerlukan imbalan tersebut setelah mereka enggan menjamu kita.’ ” (Syaikh Ahmad Syakir, 2016)

Tafsir oleh Kementrian Agama RI mengenai ayat tersebut berbunyi “Permohonan nabi musa dikabulkan oleh hamba yang saleh itu, maka keduanya berjalan meneruskan pengembaraan hingga suatu ketika keduanya sampai di suatu negeri. Mereka datang kepada penduduk setempat dan bertanya tentang negeri itu. Rasa lapar yang mendera memaksa mereka berdua meminta dijamu oleh penduduknya, tetapi mereka tidak mau menjamu mereka. Karena tidak dijamu, kemudian keduanya melanjutkan perjalanan. Tidak lama sesudah itu mereka mendapatkan dinding sebuah rumah yang hampir roboh di negeri itu. Tanpa disuruh, lalu dia, hamba yang saleh itu, menegakkannya. Dengan terheran, dia, yaitu musa, berkata kepadanya, 'jika engkau mau, niscaya engkau dapat meminta imbalan untuk pekerjaan yang telah kaulakukan itu.’ ”

Dalam konteks segmentasi kerusakan, pada ayat tersebut Nabi Khidir AS mampu mendeteksi kerusakan pada sebuah bangunan. Setelah mengetahui kerusakan pada sebuah bangunan tersebut, maka langkah yang dilakukan oleh Nabi Khididr adalah memperbaiki bangunan tersebut. Begitupula pada penelitian ini, setelah kita mengetahui kerusakan bangunan yang diakibatkan oleh bencana alam, maka yang kita lakukan adalah memperbaiki apa yang telah rusak akibat bencana alam.

Dalam Al-Quran surat Al-Baqarah ayat 11 – 12 juga berbicara mengenai kerusakan dan perbaikan :

وَإِذَا قِيلَ لَهُمْ لَا تُفْسِدُوا۟ فِى ٱلْأَرْضِ قَالُوٓا۟ إِنَّمَا نَحْنُ مُصْلِحُونَ ١١ أَلَآ إِنَّهُمْ هُمُ ٱلْمُفْسِدُونَ وَلَٰكِن لَّا يَشْعُرُون

"Dan bila dikatakan kepada mereka: "Janganlah kamu membuat kerusakan di muka bumi". Mereka menjawab: "Sesungguhnya kami orang-orang yang mengadakan perbaikan". (11) "Ingatlah, sesungguhnya mereka itulah orang-orang yang membuat kerusakan, tetapi mereka tidak sadar." (QS. Al – Baqarah: 11 – 12)

Tafsir yang merujuk pada Al-Mukhtasar untuk kedua ayat tersebut yaitu : "Apabila mereka dilarang membuat kerusakan di muka bumi berupa kekafiran, perbuatan dosa dan lain-lain, mereka mengingkarinya. Mereka beranggapan bahwa mereka adalah orang-orang yang baik dan selalu menganjurkan perbaikan. (11) Padahal sesungguhnya mereka adalah pembuat kerusakan, tetapi mereka tidak menyadarinya. Dan mereka juga tidak merasa bahwa perbuatan mereka adalah kerusakan itu sendiri. (12) " (Syaikh Ahmad Syakir, 2016)

Berikutnya tafsir oleh Kementrian Agama RI yang berbunyi "Dan apabila dikatakan dan dinasihatkan kepada mereka, janganlah berbuat kerusakan di bumi, dengan melanggar nilai-nilai yang ditetapkan agama, menghalangi orang dari jalan Allah, menyebar fitnah, dan memicu konflik, mereka justru mengklaim bahwa diri mereka bersih dari perusakan dan tidak bermaksud melakukan kerusakan. Mereka menjawab, sesungguhnya kami justru orang-orang yang melakukan perbaikan. Itu semua akibat rasa bangga diri mereka yang berlebihan. Begitulah perilaku setiap perusak yang tertipu oleh dirinya: selalu merasa kerusakan yang dilakukannya sebagai kebaikan. Karena kelakuan mereka yang selalu menampakkan keimanan dan menyembunyikan kekufuran serta

menganggap kerusakan mereka sebagai kebaikan, Allah mengingatkan orang-orang mukmin agar tidak tertipu dengan itu semua. Ingatlah, sesungguhnya merekalah yang berbuat kerusakan. Diri mereka telah rusak karena keyakinan yang batil dan perbuatan yang jahat. Mereka pun telah merusak orang lain dengan menyebar fitnah dan memicu konflik di tengah masyarakat. Tetapi, karena hati yang telah tertutup dan rasa bangga diri yang berlebihan, mereka tidak menyadari kerusakan tersebut dan akibat buruk yang akan menimpa mereka oleh sebab kemunafikan (11). Karena kelakuan mereka yang selalu menampakkan keimanan dan menyembunyikan kekufuran serta menganggap kerusakan mereka sebagai kebaikan, Allah mengingatkan orang-orang mukmin agar tidak tertipu dengan itu semua. Ingatlah, sesungguhnya merekalah yang berbuat kerusakan. Diri mereka telah rusak karena keyakinan yang batil dan perbuatan yang jahat. Mereka pun telah merusak orang lain dengan menyebar fitnah dan memicu konflik di tengah masyarakat. Tetapi, karena hati yang telah tertutup dan rasa bangga diri yang berlebihan, mereka tidak menyadari kerusakan tersebut dan akibat buruk yang akan menimpa mereka oleh sebab kemunafikan. Dan apabila dikatakan dan dinasihatkan kepada mereka, berimanlah kamu dengan tulus ikhlas sebagaimana orang lain yang menyambut suara dan seruan akal sehat telah beriman, seperti yang dilakukan para sahabat pengikut nabi Muhammad, mereka menjawab dengan penuh kesombongan dan nada menghina, apakah kami akan beriman seperti orang-orang yang kurang akal itu beriman' tidak pantas bagi kami untuk mengikuti orang-orang bodoh itu, sebab dengan begitu berarti kami sama bodohnya dengan mereka. Allah membantah kecongkakan mereka dengan

mengingatkan orangorang mukmin, ingatlah, sesungguhnya hanya mereka itulah orang-orang yang kurang akal dan bodoh, tetapi mereka tidak tahu dan tidak sadar bahwa kebodohan dan sifat kurang akal itu ada dalam diri mereka, dan mereka juga tidak menyadari kesesatan mereka itu (12).” (Kemenag RI, 2016b)

Penelitian segmentasi citra bangunan untuk menentukan tingkat kerusakan pasca bencana alam menggunakan convolutional neural network tidak hanya merupakan upaya ilmiah untuk mengukur dampak bencana, tetapi juga memiliki relevansi spiritual yang dalam. Ayat Alquran Surah Al-Baqarah (2:11-12) menegaskan pentingnya menjaga bumi dari kerusakan yang diciptakan oleh tangan manusia. Dalam konteks ini, penelitian ini menjadi suatu wujud kontribusi dalam "memperbaiki" kerusakan yang diakibatkan oleh bencana alam. Melalui segmentasi citra bangunan, kita dapat mengidentifikasi bangunan yang rusak dan merencanakan langkah-langkah pemulihan untuk mengembalikan keindahan dan fungsi bangunan tersebut, sejalan dengan nilai-nilai pemulihan lingkungan yang diajarkan dalam Alquran.

Pada ayat Al-Quran surat Ar-Rum ayat 41 juga merupakan ayat Al-Quran mengenai keruskan :

ظَهَرَ ٱلْفَسَادُ فِى ٱلْبَرِّ وَٱلْبَحْرِ بِمَا كَسَبَتْ أَيْدِى ٱلنَّاسِ لِيُذِيقَهُم بَعْضَ ٱلَّذِى عَمِلُوا۟ لَعَلَّهُمْ يَرْجِعُونَ

”Telah nampak kerusakan di darat dan di laut disebabkan karena perbuatan tangan manusia, supaya Allah merasakan kepada mereka sebahagian dari (akibat) perbuatan mereka, agar mereka kembali (ke jalan yang benar).” (QS. Ar – Rum: 41)

Tafsir ayat tersebut berdasarkan Al-Mukhtasar yang berbunyi : “Telah nampak kerusakan di daratan maupun di lautan dalam kehidupan manusia dengan berkurangnya penghasilan dan di dalam diri mereka dengan timbulnya berbagai

penyakit dan wabah, disebabkan karena kemaksiatan yang mereka lakukan. Hal itu timbul agar Allah merasakan kepada mereka balasan dari perbuatan buruk mereka di kehidupan dunia dengan harapan agar mereka kembali kepada-Nya dengan bertobat." (Syaikh Ahmad Syakir, 2016)

Berikutnya tafsir oleh Kementrian Agama RI berbunyi "Bila pada ayat-ayat sebelumnya Allah menjelaskan sifat buruk orang musyrik mekah yang menuhankan hawa nafsu, melalui ayat ini Allah menegaskan bahwa kerusakan di bumi adalah akibat mempertuhankan hawa nafsu. Telah tampak kerusakan di darat dan di laut, baik kota maupun desa, disebabkan karena perbuatan tangan manusia yang dikendalikan oleh hawa nafsu dan jauh dari tuntunan fitrah. Allah menghendaki agar mereka merasakan sebagian dari akibat perbuatan buruk mereka agar mereka kembali ke jalan yang benar dengan menjaga kesesuaian perilakunya dengan fitrahnya. 42. Perbuatan buruk manusia akan mendatangkan azab sebagaimana azab yang telah menimpa umat-umat terdahulu. Azab itu juga akan datang kepada umat-umat di masa sekarang maupun yang akan datang sebagai sunatullah jika mereka memiliki karakter yang sama. Karena itu, katakanlah, wahai nabi Muhammad, kepada siapa saja yang meragukan hakikat ini, 'bepergianlah di muka bumi, di mana saja yang bisa kamu jangkau, lalu lihatlah bagaimana kesudahan orang-orang dahulu yang dihancurkan akibat perilaku buruk mereka. Itu semua karena kebanyakan dari mereka adalah orang-orang yang mempersekutukan Allah dan menuhankan hawa nafsu. ' ". (Kemenag RI, 2016c)

Penelitian ini juga memperkuat pesan yang disampaikan dalam ayat Alquran Surah Ar-Rum (30:41) yang menggambarkan dampak buruk yang ditimbulkan oleh perilaku manusia terhadap lingkungan. Dengan menggunakan teknologi segmentasi citra bangunan, kita dapat melihat secara visual dampak destruktif yang diakibatkan oleh bencana alam, yang merupakan konsekuensi dari perilaku manusia terhadap alam. Dengan memahami dan mengukur tingkat kerusakan yang disebabkan oleh bencana alam, penelitian ini menjadi langkah awal untuk memperbaiki perilaku manusia terhadap lingkungan dan mengurangi dampak negatifnya, sejalan dengan pesan Alquran untuk menjaga dan melindungi bumi sebagai amanah dari Allah SWT.

Kisah perjalanan Nabi Musa AS dan Nabi Khidir AS dalam Al-Qur'an, khususnya dalam surat Al-Kahfi, mengilustrasikan kemampuan untuk mendeteksi dan memperbaiki kerusakan, seperti yang terlihat ketika Nabi Khidir AS memperbaiki sebuah rumah yang hampir roboh. Analogi ini menggambarkan konsep integrasi antara ayat Al-Qur'an dengan penelitian tentang segmentasi kerusakan bangunan. Seperti halnya Nabi Khidir AS yang mendeteksi dan memperbaiki kerusakan fisik, penelitian ini bertujuan untuk mendeteksi dan memperbaiki kerusakan pasca-bencana alam melalui segmentasi citra bangunan menggunakan convolutional neural network. Dengan demikian, penelitian ini bukan hanya merupakan upaya ilmiah untuk mengukur dampak bencana, tetapi juga memiliki relevansi spiritual yang dalam dengan konsep "memperbaiki" kerusakan yang diakibatkan oleh bencana alam, sebagaimana ditegaskan dalam Al-Qur'an.

Ayat-ayat Al-Qur'an, seperti yang terdapat dalam surat Al-Baqarah ayat 11-12 dan surat Ar-Rum ayat 41, memberikan panduan moral dan spiritual tentang pentingnya menjaga bumi dan dampak buruk dari perilaku manusia terhadap lingkungan. Penelitian ini menguatkan pesan-pesan tersebut dengan menghubungkan konsep kerusakan lingkungan yang diperkenalkan dalam Al-Qur'an dengan penggunaan teknologi segmentasi citra bangunan. Dengan menggunakan metode ini, penelitian mampu secara visual mengilustrasikan dampak destruktif yang diakibatkan oleh bencana alam serta perilaku manusia yang tidak bertanggung jawab terhadap alam. Melalui pemahaman yang lebih baik tentang kerusakan yang disebabkan oleh bencana alam, penelitian ini memberikan landasan bagi upaya pemulihan lingkungan dan mengurangi dampak negatifnya, sejalan dengan nilai-nilai yang diajarkan dalam Al-Qur'an tentang menjaga dan melindungi bumi sebagai amanah dari Allah SWT. Dengan demikian, integrasi antara ayat Al-Qur'an dan penelitian segmentasi kerusakan bangunan ini menjadi suatu langkah penting dalam memperbaiki hubungan manusia dengan lingkungan sesuai dengan ajaran agama.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian, *Convolutional Neural Network* (CNN) belum mampu mengukur akurasi dan *Mean Intersection over Union* (MIoU) dalam segmentasi citra bangunan rusak pasca bencana alam untuk menentukan tingkat kerusakan pada dataset ini dengan berbagai skenario yang dilakukan. Penelitian ini menghadapi tantangan signifikan terkait *overfitting* pada model *Convolutional Neural Network* yang digunakan. Hasil terbaik pada skenario 4A menunjukkan MIoU sebesar 0.31 dan akurasi sebesar 0.9577, namun kompleksitas arsitektur *Convolutional Neural Network* tidak seimbang dengan karakteristik dataset, menyebabkan *overfitting*. Meskipun demikian, *Convolutional Neural Network* menunjukkan potensi dalam memberikan prediksi tingkat kerusakan bangunan dengan akurasi yang dapat diandalkan dan memberikan nilai MIoU sebagai ukuran kualitas segmentasi. Diperlukan penyesuaian lebih lanjut dalam desain arsitektur *Convolutional Neural Network* untuk meminimalkan *overfitting* dan meningkatkan kecocokan dengan dataset yang digunakan. Kesimpulan dari penelitian ini menekankan bahwa meskipun *Convolutional Neural Network* belum optimal dalam konteks ini, tetap menjadi metode efektif dalam analisis kerusakan bangunan pasca bencana alam melalui segmentasi citra.

5.2 Saran

Setelah melakukan analisis terhadap hasil pengujian, penulis menyadari bahwa penelitian ini masih memiliki kekurangan dan memerlukan penyempurnaan lebih lanjut. Sehubungan dengan hal tersebut, penulis mengajukan beberapa rekomendasi yang dapat dipertimbangkan untuk penelitian di masa mendatang:

1. Mengguanakan data yang lebih bervariasi yaitu dengan menggunakan juga data gambar sebelum terjadinya bencana alam guna memperkaya informasi.
2. Menggunakan data selain aerial image sehingga mampu mendeteksi kerusakan bagunan lebih detail.

DAFTAR PUSTAKA

Aghalari, M., Aghagolzadeh, A., & Ezoji, M. (2021). Brain tumor image segmentation via asymmetric/symmetric UNet based on two-pathway-residual blocks. *Biomedical Signal Processing and Control*, *69*. https://doi.org/10.1016/j.bspc.2021.102841

Almais, A. T. W., Susilo, A., Naba, A., Sarosa, M., Crysdian, C., Basid, P. M. N. S. A., Hariyadi, M. A., Tazi, I., Arif, Y. M., & Wicaksono, H. (2023). SASSD: A Smart Assessment System For Sector Damage Post-Natural Disaster Using Artificial Neural Networks. *2023 2nd International Conference on Computer System, Information Technology, and Electrical Engineering (COSITE)*, 96–101. https://doi.org/10.1109/COSITE60233.2023.10249540

Annafii, M. N., Putra, O. V., Harmini, T., & Trisnaningrum, N. (2022). Segmentasi Semantik pada Citra Hama Leafblast Menggunakan Unet dan Optimasi Hyperband. *Prosiding Sains Nasional Dan Teknologi*, *12*(1), 453. https://doi.org/10.36499/psnst.v12i1.7230

Di Benedetto, A., Fiani, M., & Gujski, L. M. (2023). U-Net-Based CNN Architecture for Road Crack Segmentation. *Infrastructures*, *8*(5). https://doi.org/10.3390/infrastructures8050090

El Rai, M. C., Aburaed, N., Al-Saad, M., Al-Ahmad, H., Al Mansoori, S., & Marshall, S. (2020, November 23). Integrating deep learning with active contour models in remote sensing image segmentation. *ICECS 2020 - 27th IEEE International Conference on Electronics, Circuits and Systems, Proceedings*. https://doi.org/10.1109/ICECS49266.2020.9294806

Gangurde, S. (2023). *Building and Road Segmentation Using EffUNet and Transfer Learning Approach*. http://arxiv.org/abs/2307.03980

Goodfellow, I., Bengio, Y., & Courville, A. (n.d.). *Deep Learning*.

Irwansyah, E., Heryadi, Y., & Santoso Gunawan, A. A. (2020). Semantic Image Segmentation for Building Detection in Urban Area with Aerial Photograph Image using U-Net Models. *2020 IEEE Asia-Pacific Conference on Geoscience, Electronics and Remote Sensing Technology (AGERS)*, 48–51. https://doi.org/10.1109/AGERS51788.2020.9452773

Karki, S., & Kulkarni, S. (2021). Ship detection and segmentation using Unet. *Proceedings of the 2021 1st International Conference on Advances in Electrical, Computing, Communications and Sustainable Technologies, ICAECT 2021*. https://doi.org/10.1109/ICAECT49130.2021.9392463

Kemenag RI. (2016a). *Tafsir Wajiz I* (1st ed., Vol. 1). Lajnah Pentashihan Mushaf Al-Qur'an.

Kemenag RI. (2016b). *Tafsir Wajiz II* (1st ed., Vol. 2). Lajnah Pentashihan Mushaf Al-Qur'an.

Kemenag RI. (2016c). *Tafsir Wajiz II* (1st ed., Vol. 2). Lajnah Pentashihan Mushaf Al-Qur'an.

Khan, A., Sohail, A., Zahoora, U., & Qureshi, A. S. (2019). *A Survey of the Recent Architectures of Deep Convolutional Neural Networks*. https://doi.org/10.1007/s10462-020-09825-6

Khan, M. Z., Gajendran, M. K., Lee, Y., & Khan, M. A. (2021). Deep Neural Architectures for Medical Image Semantic Segmentation: Review. *IEEE Access*, *9*, 83002–83024. https://doi.org/10.1109/ACCESS.2021.3086530

Khan, W. A., Chung, S. H., Awan, M. U., & Wen, X. (2020). Machine learning facilitated business intelligence (Part I): Neural networks learning algorithms and applications. *Industrial Management and Data Systems*, *120*(1), 164–195. https://doi.org/10.1108/IMDS-07-2019-0361

Kotaridis, I., & Lazaridou, M. (2022). SEMANTIC SEGMENTATION USING A UNET ARCHITECTURE ON SENTINEL-2 DATA. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *43*(B3-2022), 119–126. https://doi.org/10.5194/isprs-archives-XLIII-B3-2022-119-2022

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Nascimento, V. F., Sobral, A. C., Andrade, P. R., Yesiller, N., & Ometto, J. P. H. B. (2017). Natural Disaster Risk in Municipal Solid Waste Disposal Sites Using GIS: A Case Study in São Paulo State, Brazil. *Journal of Water Resource and Protection*, *09*(11), 1213–1224. https://doi.org/10.4236/jwarp.2017.911079

Ojha, V. K., Dutta, P., Saha, H., & Ghosh, S. (2012). Detection of Proportion of Different Gas Components Present in Manhole Gas Mixture Using Backpropagation Neural Network. *Technology*, *37*(Icint), 11–15.

Padli, J., Habibullah, M. S., & Baharom, A. H. (2010). Economic impact of natural disasters' fatalities. *International Journal of Social Economics*, *37*(6), 429–441. https://doi.org/10.1108/03068291011042319

Pahlavani, P., Samadzadegan, F., & Delavar, M. R. (2006). *A GIS-Based*

Approach for Urban Multi-criteria Quasi Optimized Route Guidance by Considering Unspecified Site Satisfaction (pp. 287–303). https://doi.org/10.1007/11863939_19

Pramono, A., Mahendra, I. W. S., Wijaya, I. B. A., Agustina, I. A., & Herman, R. T. (2022). Diagnosis and Repair of the Cracking House next to the River. *IOP Conference Series: Earth and Environmental Science*, *998*(1). https://doi.org/10.1088/1755-1315/998/1/012002

Rehan, R., Bastian, E., & Kurniawan, D. (2023). EVALUASI STRUKTUR BANGUNAN SEKOLAH PASCA GEMPA DI MTS MUHAMMADIYAH KAJAI KABUPATEN PASAMAN BARAT. *Rang Teknik Journal*, *6*(2), 218–224. https://doi.org/10.31869/rtj.v6i2.4205

Rosselló, J., Becken, S., & Santana-Gallego, M. (2020). The effects of natural disasters on international tourism: A global analysis. *Tourism Management*, *79*(April 2019). https://doi.org/10.1016/j.tourman.2020.104080

Sivagami, S., Chitra, P., Kailash, G. S. R., & Muralidharan, S. R. (2020). UNet Architecture Based Dental Panoramic Image Segmentation. *2020 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, 187–191. https://doi.org/10.1109/WiSPNET48689.2020.9198370

Sreekumar, A., & Geetha, M. (2020). Hand Segmentation In Complex Background Using UNet. *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, 440–445. https://doi.org/10.1109/ICIRCA48905.2020.9183215

Strohmann, T., Bugelnig, K., Breitbarth, E., Wilde, F., Steffens, T., Germann, H., & Requena, G. (2019). Semantic segmentation of synchrotron tomography of multiphase Al-Si alloys using a convolutional neural network with a pixel-wise weighted loss function. *Scientific Reports*, *9*(1), 1–10. https://doi.org/10.1038/s41598-019-56008-7

Syaikh Ahmad Syakir. (2016). *Mukhtashar tafsir Ibnu katsir* (1st ed., Vol. 1). Darus Sunnah Press.

Zaryabi, E. H., Kalantar, B., Moradi, L., Halin, A. A., & Ueda, N. (2022). MSBDA-Net: Multi-scale Siamese Building Damage Assessment Network. *2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, 1–6. https://doi.org/10.1109/CSDE56538.2022.10089353