

SKRIPSI

**IMPLEMENTASI *DEEPLABV3+* UNTUK PENINGKATAN
DETEKSI DAN *TRACKING* LAJUR JALAN PADA SISTEM
*AUTONOMOUS CAR***

Disusun dan diajukan oleh:

**A. ICHSAN MUDATSIR
D121 20 1009**



**PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
GOWA
2024**

LEMBAR PENGESAHAN SKRIPSI

**IMPLEMENTASI *DEEPLABV3+* UNTUK PENINGKATAN
DETEKSI DAN *TRACKING* LAJUR JALAN PADA SISTEM
*AUTONOMOUS CAR***

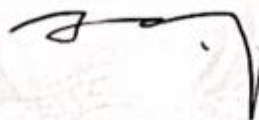
Disusun dan diajukan oleh

**A. Ichsan Mudatsir
D121201009**

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian
Studi Program Sarjana Program Studi Teknik Informatika
Fakultas Teknik Universitas Hasanuddin
Pada tanggal 21 Agustus 2024
dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui,

Pembimbing Utama,



Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM.ASEAN.Eng.
NIP 19750716 200212 1 004

Ketua Program Studi,



Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM.ASEAN.Eng.
NIP 19750716 200212 1 004

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini ;
Nama : A. Ichsan Mudatsir
NIM : D121201009
Program Studi : Teknik Informatika
Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

Implementasi DeepLabv3+ Untuk Peningkatan Deteksi dan Tracking Lajur Jalan Pada Sistem Autonomous car

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain dan bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Semua informasi yang ditulis dalam skripsi yang berasal dari penulis lain telah diberi penghargaan, yakni dengan mengutip sumber dan tahun penerbitannya. Oleh karena itu semua tulisan dalam skripsi ini sepenuhnya menjadi tanggung jawab penulis. Apabila ada pihak manapun yang merasa ada kesamaan judul dan atau hasil temuan dalam skripsi ini, maka penulis siap untuk diklarifikasi dan mempertanggungjawabkan segala resiko.

Segala data dan informasi yang diperoleh selama proses pembuatan skripsi, yang akan dipublikasi oleh Penulis di masa depan harus mendapat persetujuan dari Dosen Pembimbing.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan isi skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 21 Agustus 2024

Yang Menyatakan



A. Ichsan Mudatsir

ABSTRAK

A. ICHSAN MUDATSIR. *Implementasi DeepLabv3+ Untuk Peningkatan Deteksi dan Tracking Lajur Jalan Pada Sistem Autonomous car (dibimbing oleh Indrabayu)*

Latar belakang. Proyeksi pasar kendaraan otonom diperkirakan mencapai \$615 miliar pada 2026, mencerminkan pergeseran industri dari kecepatan ke keselamatan dan inovasi teknologi. Salah satu aspek keselamatan jalan raya adalah mendeteksi lajur jalan dengan akurat. Penelitian sebelumnya menggunakan arsitektur U-Net dalam segmentasi menunjukkan akurasi keseluruhan 79,8%, dengan akurasi garis putus-putus 75,3% dan garis penuh 54,8%. Namun, U-Net memiliki kelemahan dalam segmentasi karena fitur garis yang mirip dan segmentasi dilakukan per *frame*. Oleh karena itu, penerapan *DeepLabv3+* dapat meningkatkan akurasi dari peneliti sebelumnya. **Tujuan.** Penelitian ini bertujuan untuk merancang dan mengoptimalkan sistem pendeteksi dan *tracking* lajur jalan pada *Autonomous car* menggunakan arsitektur *DeepLabv3+* dan menganalisis performa *Semantic Segmentation* dalam mendeteksi dan *tracking* lajur jalan menggunakan arsitektur *DeepLabv3+*. **Metode.** Penelitian ini menggunakan data sebanyak 374 gambar berupa lingkungan jalan diantaranya Jl. Veteran Utara, Jl. Gunung Bawakaraeng, Jl. Tol Reformasi, Jl. Tol Layang A.P Pettarani, dan Jl. Bonto Daeng Ngirate Kota Makassar terdiri dari empat kelas yakni jalan, marka garis membujur penuh, garis membujur putus-putus dan *background*. Data dibagi menjadi 336 data untuk *training* dan 38 untuk *testing*. Kemudian dilakukan proses *training* berupa *hyperparameter* dan berbagai metode *training* lainnya, kemudian dilakukan evaluasi model menggunakan *Intersection Over Union* (IoU) dan perhitungan estimasi jarak ban dengan marka menggunakan *Root Mean Square Error* (RMSE). **Hasil.** Pengujian model menggunakan IoU mencapai akurasi sebesar 97,34% dan hasil pengujian sistem estimasi jarak ban dan marka garis dengan metode RMSE sebesar 0,0377119856. **Kesimpulan.** Implementasi *Deeplabv3+* meningkatkan hasil IOU secara keseluruhan sehingga meningkatkan akurasi deteksi lajur jalan.

Kata Kunci: *Autonomous car*, *real-time*, lajur jalan, *Semantic Segmentation*, *DeepLabv3+*

ABSTRACT

A. ICHSAN MUDATSIR. *Implementation of DeepLabv3+ for Improved Lane Detection and Tracking in Autonomous Car System* (supervised by Indrabayu)

Background. The projected autonomous vehicle market is expected to reach \$615 billion by 2026, reflecting the industry's shift from speed to safety and technological innovation. One aspect of road safety is accurately detecting road lanes. Previous research using U-Net architecture in segmentation showed an overall accuracy of 79,8%, with dashed line accuracy of 75,3% and full line 54,8%. However, U-Net has weaknesses in segmentation due to similar line features and segmentation is done per frame. Therefore, the application of DeepLabv3+ can improve the accuracy from previous researchers. **Objective.** This research aims to design and optimize the road lane detection and tracking system in Autonomous car using DeepLabv3+ architecture and analyze the performance of Semantic Segmentation in detecting and tracking road lanes using DeepLabv3+ architecture. **Methods.** This study uses data as many as 374 images in the form of road environments including Jl. Veteran Utara, Jl. Gunung Bawakaraeng, Jl. Tol Reformasi, Jl. Tol Layang A.P Pettarani, dan Jl. Bonto Daeng Ngirate Makassar City consisting of four classes namely roads, full longitudinal line markings, broken longitudinal lines and backgrounds. The data is divided into 336 data for training and 38 for testing. Then the training process is carried out in the form of hyperparameters and various other training methods, then model evaluation using Intersection Over Union (IoU) and calculation of tire distance estimation with markings using Root Mean Square Error (RMSE). **Result.** Model testing using IoU achieved an accuracy of 97,34% and the results of testing the tire distance estimation system and line markings with the RMSE method were 0,0377119856. **Conclusion.** The implementation of Deeplabv3+ improves the overall IOU results thus improving the accuracy of road lane detection.

Keywords: Autonomous car, real-time, road lane, Semantic Segmentation, DeepLabv3+

DAFTAR ISI

LEMBAR PENGESAHAN SKRIPSI.....	i
PERNYATAAN KEASLIAN.....	ii
ABSTRAK	iii
ABSTRACT	iv
DAFTAR ISI.....	v
DAFTAR GAMBAR	vii
DAFTAR TABEL.....	ix
DAFTAR SINGKATAN DAN ARTI SIMBOL	x
DAFTAR LAMPIRAN.....	xi
KATA PENGANTAR	xii
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	4
1.3 Tujuan Penelitian	4
1.4 Manfaat Penelitian	4
1.5 Ruang Lingkup Penelitian.....	4
BAB II TINJAUAN PUSTAKA.....	6
2.1 Marka Jalan	6
2.2 <i>Autonomous car</i>	10
2.3 Visi Komputer.....	13
2.4 <i>Semantic Segmentation</i>	14
2.5 <i>Convolutional Neural Network (CNN)</i>	16
2.6 <i>Fully Convolutional Network For Semantic Segmentation</i>	22
2.7 <i>DeepLabv3+</i>	24
2.8 ResNet 18 Backbone.....	29
2.9 <i>Convolutional Block Attention Module (CBAM)</i>	31
2.10 <i>Squeeze-and-Excitation (SE) Block</i>	33
2.11 <i>Efficient channel attention (ECA) Layer</i>	36
2.12 <i>Feature Cross Attention (FCA)</i>	38
2.13 <i>Feature Enhancement Module (FEM)</i>	41
2.14 <i>Loss Function</i>	42
2.15 <i>Inverse Perspective Mapping (IPM)</i>	43
2.16 <i>Intersection over Union (IoU)</i>	46
2.17 <i>Root-Mean-Square Error (RMSE)</i>	48
BAB III METODE PENELITIAN/PERANCANGAN	50
3.1 Waktu dan Lokasi Penelitian	50
3.2 Benda Uji dan Alat.....	50
3.3 Tahapan Penelitian	50
3.4 Teknik Pengambilan Data.....	52
3.5 Perancangan Sistem	53
3.6 Analisis Kerja Sistem.....	63
BAB IV Hasil dan pembahasan	67
4.1 Hasil Penelitian	67
4.2 Pembahasan.....	74
BAB V Kesimpulan dan saran	80

5.1 Kesimpulan	80
5.2 Saran.....	80
DAFTAR PUSTAKA	81

DAFTAR GAMBAR

Gambar 1 Peraturan marka garis bujur penuh.....	7
Gambar 2 Marka garis membujur putus-putus pada jalan 2 jalur, sesuai dengan peraturan Menteri Perhubungan	8
Gambar 3 Marka garis membujur putus-putus pada jalan 2 jalur, sesuai dengan peraturan Menteri Perhubungan	8
Gambar 4 Marka garis membujur kombinasi penuh dan putus-putus	9
Gambar 5 Warna marka: (a) membujur jalan Nasional; (b) garis membujur jalan selain jalan Nasional.....	10
Gambar 6 SAE's <i>levels of driving automation</i>	12
Gambar 7 Perbedaan tugas visi komputer.....	14
Gambar 8 Semantic Segmentation secara Holistic	16
Gambar 9 Lapisan arsitektur dalam CNN.....	17
Gambar 10 Nilai gambar dalam bentuk matrix.....	18
Gambar 11 Convolutional Layer.....	19
Gambar 12 Contoh <i>max pooling</i> pada CNN	21
Gambar 13 <i>Fully Connected Layer</i>	22
Gambar 14 Arsitektur FCN.....	23
Gambar 15 Arsitektur <i>Encoder-Decoder</i> mendetail dari <i>DeepLabv3+</i> untuk <i>semantic segmentation</i>	24
Gambar 16 <i>Atrous convolution</i> dengan kernel 3x3 dan rate yang berbeda.....	25
Gambar 17 Konvolusi: (a) tanpa <i>atrous convolution</i> ; (b) dengan <i>atrous convolution</i>	27
Gambar 18 <i>Atrous Spatial Pyramid Pooling (ASPP)</i>	27
Gambar 19 <i>Residual Block</i>	29
Gambar 20 Skema diagram ResNet-18 backbone: (a) <i>Basic Block 1</i> ; (b) <i>Basic Block 2</i> ; (c) Resnet 18 Backbone	30
Gambar 21 Struktur CBAM: (a) <i>Channel Attention Module</i> ; (b) <i>Spatial Attention Module</i> ; (c) CBAM.....	32
Gambar 22 <i>Squeeze-and-Excitation block</i>	33
Gambar 23 Penerapan SE block pada modul Inception (kiri) dan ResNet (kanan).....	35
Gambar 24 <i>Efficient channel attention module</i>	37
Gambar 25 <i>Feature Cross Attention Module (FCA)</i>	38
Gambar 26 <i>Channel attention module (CA)</i>	39
Gambar 27 <i>Spatial attention module (SA)</i>	40
Gambar 28 Arsitektur <i>Feature enchainement module (FEM)</i>	41
Gambar 29 IPM: a) <i>input</i> gambar jalanan; (b) <i>output</i> gambar jalan setelah penerapan IPM	44
Gambar 30 Implementasi IPM	46
Gambar 31 Ilustrasi skema IoU.....	47
Gambar 32 Contoh hasil perhitungan IoU	48
Gambar 33 Diagram tahapan penelitian.....	51
Gambar 34 Lokasi <i>Dashcam</i>	53
Gambar 35 Flowchart perancangan sistem	54
Gambar 36 Contoh anotasi gambar menggunakan Labelme	55

Gambar 37 Contoh Dataset dengan standar Pascal VOC	56
Gambar 38 Arsitektur <i>DeepLabv3+</i>	58
Gambar 39 Contoh IPM	60
Gambar 40 Flowchart <i>Lane Departure Warning System</i>	60
Gambar 41 Hasil gambar IPM yang tersegmentasi.....	61
Gambar 42 Ilustrasi penentuan koordinat ban	62
Gambar 43 Gambar: (a) Aktual; (b) Prediksi; (c) (Aktual \cap Prediksi) (d) (Aktual \cup Prediksi)	64
Gambar 44 Skenario: (a) Skenario 1 (b) Skenario 2 (c) Skenario 3 (d) Skenario 4 (e) Skenario 5	65
Gambar 45 Visualisasi Gambar; (a) Citra Input,(b) Citra Segmentasi Aktual, (c) Prediksi	68
Gambar 46 Hasil segmentasi pada saat pengujian Real-time	69
Gambar 47 Contoh hasil salah segmentasi pada kondisi hujan	77
Gambar 48 Contoh hasil salah segmentasi pada kondisi malam hari	78
Gambar 49 Grafik jarak dan estimasi jarak pada sistem.....	78

DAFTAR TABEL

Tabel 1. Hasil Segmentasi (IoU) pada <i>PASCAL VOC 2012 dataset</i>	28
Tabel 2. Hasil Segmentasi (IoU) pada <i>cityscapes dataset</i>	29
Tabel 3. Detail ResNet-18 Backbone.....	31
Tabel 4. Label encode tiap kelas	57
Tabel 5. Tabel pengujian.....	66
Tabel 6. Akurasi tiap kelas.....	67
Tabel 7. Hasil Skenario 1	69
Tabel 8. Hasil Skenario 2.....	70
Tabel 9. Hasil Skenario 3.....	71
Tabel 10. Hasil Skenario 4.....	72
Tabel 11. Hasil Rata-rata Perhitungan <i>Root Mean Square Error</i> setiap pengujian	73

DAFTAR SINGKATAN DAN ARTI SIMBOL

Lambang/Singkatan	Arti dan Keterangan
M	<i>Meter</i>
Cm	<i>Centimeter</i>
Km	<i>Kilometer</i>
Σ	<i>Sigma</i>
MLP	<i>Multilayer perception</i>
d_E	<i>Euclidean Distance</i>
σ	<i>Fungsi aktivasi sigmoid</i>
Cat	<i>Concatination</i>
Fl	<i>Focal Loss</i>
De	<i>Dice Loss</i>
Ce	<i>Cross Entropi loss</i>
IPM	<i>Invers Perspective Mapping</i>
RMSE	<i>Root Mean Squared Error</i>

DAFTAR LAMPIRAN

Lampiran 1 Contoh Dataset	86
Lampiran 2 Lembar perbaikan skripsi	88

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat Allah SWT atas berkat rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir ini yang berjudul “Implementasi *Deeplabv3+* Untuk Peningkatan Deteksi dan *Tracking* Lajur Jalan Pada Sistem *Autonomous Car*” sebagai salah satu persyaratan yang harus dipenuhi dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin. Sholawat serta salam kepada nabi Muhammad SAW yang telah menunjukkan dan mengajarkan akhlak mulia sehingga didapatkan kenyamanan dan keramahan dalam berhubungan dengan orang di sekitar.

Dengan segala kerendahan hati, penulis menyadari bahwa dalam penyusunan dan penulisan laporan tugas akhir ini tidak lepas dari bantuan, bimbingan serta dukungan dari berbagai pihak, dari masa perkuliahan sampai dengan masa penyusunan tugas akhir ini. Sehingga, pada kesempatan ini penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT. Atas semua karunia serta pertolongan-Nya yang tiada batas, yang telah diberikan kepada penulis di setiap langkah dalam penelitian hingga penulisan laporan ini.
2. Kedua orang tua penulis, Bapak A. Ruddin, S. Sos., dan Ibu Hj. Besse Nadirah, S.Pd., M.Pd. yang selalu mendoakan untuk kebaikan penulis, selalu memberikan kasih sayang, cinta, dukungan dan motivasi tidak hanya moril maupun materil. Terima kasih Bapak dan Ibu yang selalu sabar dan semangat tiada henti dalam menghadapi dan mendidik penulis.
3. Bapak Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM., ASEAN.Eng., selaku pembimbing yang senantiasa meluangkan waktu, tenaga dan pikiran serta perhatian yang luar biasa untuk membimbing penulis dalam proses penyelesaian tugas akhir ini.
4. Segenap Dosen dan Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah memberikan ilmu dan pengetahuan baru, serta bantuan kepada penulis selama menuntut masa perkuliahan.
5. Kepada Anugrah Nur Auliani yang telah menemani dan membantu peneliti dari awal perkuliahan hingga proses penyelesaian tugas akhir.
6. Segenap keluarga AIMP Research Group Universitas Hasanuddin, terutama Kak Clara Diva, Kak Herlina Anwar, kak Ijlal Nurhadi, kak Sabda, Aldilah Rezki, Nurza Zahira, Agunawan Ali Nur dan Zid Irsyadin yang telah memberikan banyak bantuan selama penelitian dan teman diskusi terkait progres penyusunan tugas akhir.
7. Nur Islamiah, Muh Thoriq dan segenap keluarga REZOLVER yang banyak membantu selama kuliah dan dalam proses menyelesaikan tugas akhir ini.

8. Serta pihak-pihak lain yang tidak sempat disebutkan dan tanpa sadar telah membantu penulis dalam menyelesaikan tugas akhir.

Dengan rasa syukur dan kerendahan hati, penulis memberikan rasa hormat yang tak terhingga, semoga Allah SWT. membalas semua kebaikan dari semua pihak yang telah banyak membantu penulis dalam menyelesaikan tugas akhir ini. Penulis menyadari tugas akhir ini masih banyak kekurangan dan jauh dari kata sempurna. Oleh karena itu, penulis mengharapkan segala masukan dan saran yang membangun sehingga tugas akhir ini dapat memberi manfaat bagi penulis dan pembaca. Akhir kata, semoga tugas akhir ini dapat dijadikan sebagai sumber ilmu pengetahuan dan bermanfaat bagi penulis dan pembaca pada umumnya.

Makassar, Agustus 2024

Penulis

BAB I PENDAHULUAN

1.1 Latar Belakang

Perkembangan industri otomotif belakangan ini mengalami kemajuan pesat dan telah menarik perhatian baik dari kalangan akademisi maupun industri. Saat ini, kita menyaksikan berbagai inovasi yang bertujuan agar kendaraan mampu bergerak secara otomatis tanpa keterlibatan manusia. Perubahan fokus inovasi dalam industri otomotif bukan lagi hanya terkait dengan aspek kecepatan, melainkan lebih berorientasi pada isu keselamatan. Nilai proyeksi untuk *autonomous vehicle* di pasar global diestimasi mencapai \$615 miliar pada tahun 2026, menandakan tren meningkatnya permintaan dan adopsi teknologi *autonomous vehicle*. Ini merefleksikan pergeseran penting dalam prioritas industri dari kecepatan ke keselamatan dan inovasi teknologi. Menurut *Transportation's National Highway Traffic Safety Administration* (NHTSA) Departemen Transportasi AS, kendaraan yang sepenuhnya otomatis atau otonom didefinisikan sebagai "kendaraan yang beroperasi tanpa masukan langsung dari pengemudi untuk mengendalikan kemudi, akselerasi, dan pengereman, dirancang sedemikian rupa sehingga pengemudi tidak diharapkan untuk terus-menerus memantau jalan selama beroperasi dalam mode *self-driving*" (Padmaja et al., 2023).

Kecelakaan di jalan raya merupakan permasalahan serius di Indonesia. Risiko kecelakaan dapat disebabkan karena aktivitas mengemudi dalam jangka waktu yang lama sehingga mengakibatkan kelelahan dan penurunan konsentrasi (Zheng et al., 2022). Sangat banyak kecelakaan terjadi akibat dari kelalaian manusia sendiri. Tercatat 61% penyebab kecelakaan terbesar karena kemampuan serta karakter pengemudi, 9% disebabkan karena faktor kendaraan (terkait dengan pemenuhan persyaratan teknis layak jalan) dan 30% disebabkan oleh faktor prasarana dan lingkungan (Marroli, 2017).

Seiring dengan kemajuan teknologi terbaru dalam bidang *DeepLearning* dan *Image Processing*, kerangka kerja *network-based neural* dapat dikembangkan untuk mendukung sistem *self driving*. Dengan tuntutan akan

keamanan dan pengemudi otomatisasi secara *real-time*, deteksi jalur menjadi poin kritis, bertujuan untuk secara otomatis mengidentifikasi garis jalan (Zheng et al., 2022).

Salah satu aspek kunci dari keselamatan jalan raya adalah kemampuan untuk mendeteksi dan mengidentifikasi jalur jalan dengan efisien dan akurat. Deteksi jalur jalan merupakan tugas khusus yang esensial untuk teknologi *autonomous driving*, yang melibatkan identifikasi batas-batas jalan dan perkiraan lintasan kendaraan dalam batas-batas tersebut. Dengan deteksi jalur jalan yang akurat, sistem bantuan pengemudi canggih dapat memberikan peringatan kepada pengemudi ketika mereka menyimpang dari jalur, dan *autonomous vehicle* dapat menjalankan navigasi dengan aman tanpa campur tangan manusia (Pandey et al., 2023).

Berdasarkan latar belakang yang telah diberikan dan untuk memperluas penelitian Nublan Azqalani Muis, penelitian ini menyajikan judul “Implementasi *DeepLabv3+* Untuk Peningkatan Deteksi dan *Tracking* Lajur Jalan Pada Sistem *Autonomous car*”, ditujukan untuk meningkatkan akurasi deteksi dengan pendekatan arsitektur yang berbeda. Penelitian ini bertujuan untuk meningkatkan akurasi deteksi lajur jalan pada sistem *autonomous car* dengan mengadopsi arsitektur *DeepLabv3+*. Pengembangan ini muncul sebagai respons terhadap hasil sebelumnya yang menggunakan arsitektur U-Net, mencapai akurasi 79,8% melalui *Intersection Over Union* (IoU). Meskipun mencapai tingkat akurasi yang signifikan, penelitian sebelumnya menghadapi tantangan dalam akurasi setiap kelas, khususnya pada garis membujur penuh dan garis membujur putus putus yang hanya mencapai 54,8% dan 75,3% (Muis, 2022). Penelitian sebelumnya juga menunjukkan adanya beberapa kesalahan segmentasi.

Kesalahan segmentasi yang teridentifikasi menjadi motivasi utama pengadopsian *DeepLabv3+*. Penelitian ini mengeksplorasi kelebihan arsitektur *DeepLabv3+* dalam mengatasi sejumlah kelemahan yang muncul pada U-Net. Salah satu kelemahan utama arsitektur U-Net yang terungkap dalam penelitian sebelumnya adalah pendekatannya yang melakukan prediksi segmentasi berdasarkan *frame per frame* (Muis, 2022), tanpa mempertimbangkan

informasi temporal antara *frame* yang berurutan. Hal ini dapat mengakibatkan ketidakkonsistenan dalam hasil segmentasi.

Seiring dengan itu, *DeepLabv3+* membawa kontribusi signifikan untuk mengatasi kekurangan ini. Kemampuannya untuk mempertimbangkan informasi temporal antara *frame* yang berurutan, didorong oleh penggunaan *Atrous convolution* dalam arsitektur, memungkinkan model untuk memeriksa detail halus dalam gambar. Ini termasuk tekstur atau fitur halus pada objek, yang esensial untuk deteksi dan klasifikasi objek dengan akurasi tinggi. Pendekatan ini menjadi krusial terutama dalam situasi dimana keberlanjutan visual antara *frame* memiliki dampak signifikan, seperti dalam penanganan lajur jalan pada lingkungan lalu lintas. *Atrous convolution* menjadi alat yang efektif, memungkinkan model untuk memperluas jangkauan pengamatan tanpa mengorbankan resolusi, sehingga meningkatkan kemampuan untuk mengenali pola-pola halus dan kontinuitas temporal pada gambar (Chen et al., 2018).

Selain dari aspek temporal, penelitian sebelumnya juga menyoroti kelemahan U-Net dalam situasi dimana garis membujur penuh dan garis membujur putus-putus memiliki fitur yang sama, menyebabkan kesalahan segmentasi (Muis, 2022). *DeepLabv3+* merespon permasalahan ini dengan memanfaatkan *Spatial Pyramid Pooling*, sebuah fitur yang memberikan model kemampuan untuk mengenali objek dalam berbagai ukuran dan posisi. Hal ini menciptakan fleksibilitas yang sangat diperlukan untuk mengatasi variasi kondisi lingkungan di jalan (Chen et al., 2017). Penerapan *DeepLabv3+* dalam penelitian ini diharapkan dapat menghadirkan peningkatan yang signifikan dalam akurasi deteksi lajur jalan pada sistem *autonomous car*. Hasil eksperimen ini diharapkan dapat memberikan kontribusi yang berharga pada perkembangan teknologi *autonomous car*, khususnya dalam konteks peningkatan keandalan dan ketepatan sistem navigasi otonom.

1.2 Rumusan Masalah

Penelitian ini berfokus pada masalah utama sebagai berikut:

1. Bagaimana mengoptimalkan hasil deteksi dan *tracking* lajur jalan pada *Autonomous car* menggunakan arsitektur *DeepLabv3+*?
2. Bagaimana menganalisis performa *Semantic Segmentation* dalam mendeteksi dan *tracking* lajur jalan menggunakan arsitektur *DeepLabv3+*?

1.3 Tujuan Penelitian

Mengacu pada masalah yang telah diidentifikasi sebelumnya, tujuan penelitian ini adalah:

1. Merancang dan mengoptimalkan sistem pendeteksi dan *tracking* lajur jalan pada *Autonomous car* menggunakan arsitektur *DeepLabv3+*.
2. Menganalisis performa *Semantic Segmentation* dalam mendeteksi dan *tracking* lajur jalan menggunakan arsitektur *DeepLabv3+*.

1.4 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Akademisi:

Diharapkan dapat menjadi referensi bagi akademisi maupun peneliti berikutnya dalam melakukan penelitian yang terkait dengan tema yang penulis teliti

2. Industri otomotif:

- Adanya sistem pendeteksi lajur jalan diharapkan dapat menurunkan risiko terjadinya kecelakaan yang disebabkan oleh manusia.
- Memberikan informasi penelitian dalam pengembangan *Autonomous car*, khususnya terkait deteksi lajur jalan selanjutnya.

1.5 Ruang Lingkup Penelitian

Adapun ruang lingkup dalam penelitian ini adalah sebagai berikut:

1. Kelas obyek yang dideteksi adalah jalan, marka garis sambung dan marka garis putus.
2. Waktu penelitian dilakukan di siang hari dengan kondisi cuaca cerah.
3. Kamera yang digunakan berupa kamera *dashcam* yang berjumlah satu buah dan diposisikan menghadap ke depan dengan sudut $89,27^\circ$.
4. Jenis mobil yang dipakai yaitu mobil *low-MPV (Multi Purpose Vehicle)*.
5. Kecepatan kendaraan pada saat pengukuran estimasi jarak antara marka dengan ban yaitu 10 km/jam.

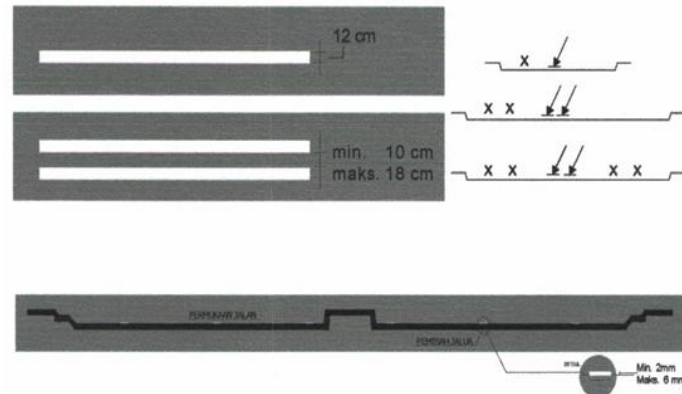
BAB II TINJAUAN PUSTAKA

2.1 Marka Jalan

Marka jalan merupakan elemen penting dalam sistem lalu lintas yang berfungsi untuk mengarahkan dan memberi informasi kepada pengendara maupun pejalan kaki. Di Indonesia, regulasi terkait marka jalan diatur dalam Peraturan Menteri Perhubungan Nomor 34 Tahun 2014, yang kemudian mengalami perubahan dengan dikeluarkannya Peraturan Menteri Perhubungan Nomor 67 Tahun 2018. Menurut regulasi tersebut, marka jalan didefinisikan sebagai berbagai jenis tanda yang terletak di permukaan jalan, meliputi garis-garis membujur, garis melintang, garis serong, dan berbagai simbol lainnya. Tujuan utama marka jalan adalah memberikan arahan dalam arus lalu lintas dan menetapkan batas-batas area tertentu dalam lalu lintas. Dokumen regulasi ini juga merinci berbagai jenis marka jalan yang digunakan untuk meningkatkan keamanan dan kelancaran lalu lintas di jalan raya. Berikut jenis-jenis marka jalan:

2.1.1 Marka Garis Membujur Penuh

Marka Garis Membujur penuh berfungsi sebagai batas yang melarang pengemudi untuk melintasinya. Tempat pemasangannya biasanya dipilih di area berbahaya seperti tikungan, tanjakan, turunan, atau daerah ramai. Tujuannya adalah untuk meningkatkan keselamatan di jalan raya dengan memberikan peringatan dan membatasi akses di lokasi-lokasi yang berpotensi membahayakan (Menteri Perhubungan Republik Indonesia, 2018).



Gambar 1 Peraturan marka garis bujur penuh
 Sumber: (Menteri Perhubungan Republik Indonesia, 2018)

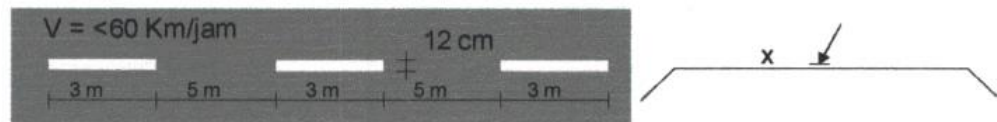
Berdasarkan gambar 1 yang menunjukkan peraturan marka garis bujur penuh, garis-garis ini terdiri dari dua garis sejajar dengan lebar masing-masing 12 cm dan jarak antar garis antara 10 cm hingga 18 cm, serta memiliki tebal antara 2 mm hingga 6 mm. Marka ini berfungsi sebagai tanda pembatas yang melarang pengemudi untuk melintasi garis tersebut, umumnya digunakan di area rawan kecelakaan atau jalur dengan kecepatan tinggi. Tujuannya adalah untuk meningkatkan keselamatan dan kelancaran lalu lintas dengan mengatur arus kendaraan agar tetap berada di jalurnya dan memberikan panduan visual yang jelas, terutama di jalan-jalan yang berbahaya seperti tikungan tajam atau tanjakan, sesuai dengan peraturan Menteri Perhubungan Republik Indonesia tahun 2018.

2.1.2 Marka Garis Membujur Putus-putus

Pengemudi diperbolehkan untuk melintasi marka ini, contohnya untuk melakukan pindah jalur atau mendahului kendaraan lain. Hal ini berlaku pada kondisi-kondisi tertentu, seperti pada jalan dengan lebar lebih dari 550 cm dan memiliki dua jalur dengan arah berlawanan dan terdapat aturan khusus yang mengatur pemasangan marka garis membujur putus-putus yang harus dipatuhi (Menteri Perhubungan Republik Indonesia, 2018).

a. Jalan 2 jalur, 2 arah dengan lebar > 550cm

Gambar 2 merupakan penerapan marka garis membujur putus-putus yang sesuai dengan peraturan Menteri Perhubungan (Permenhub) tahun 2018. Marka ini berfungsi untuk memisahkan lajur dengan arah berlawanan pada jalan dua jalur dengan lebar lebih dari 550 cm.



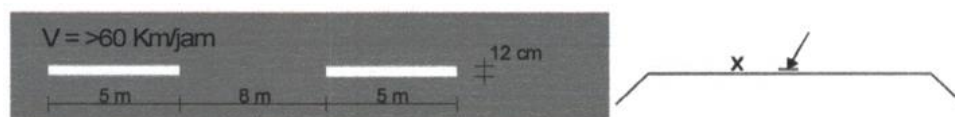
Gambar 2 Marka garis membujur putus-putus pada jalan 2 jalur, sesuai dengan peraturan Menteri Perhubungan

Sumber: (Menteri Perhubungan Republik Indonesia, 2018)

Marka pada gambar 2 digunakan pada jalan dengan batas kecepatan di bawah 60 km/jam dan memiliki dimensi garis yang berjarak 5 m antar garis putus dengan lebar garis 12 cm dan panjang gap antar garis sebesar 3 m. Marka ini memberi panduan kepada pengemudi bahwa mereka diperbolehkan untuk melintasi atau mengganti jalur di bawah kondisi lalu lintas yang ditentukan, dengan tujuan utama untuk memberikan navigasi yang aman dan jelas pada jalan raya, serta membantu mengatur arus lalu lintas yang lancar.

b. Jalan lebih dari 2 jalur

Gambar 3 menunjukkan marka garis membujur putus-putus sesuai peraturan Menteri Perhubungan 2018 untuk jalan dua jalur. Marka ini dirancang agar pengemudi dapat dengan aman berpindah jalur atau mendahului kendaraan di jalan dengan kecepatan di atas 60 km/jam.



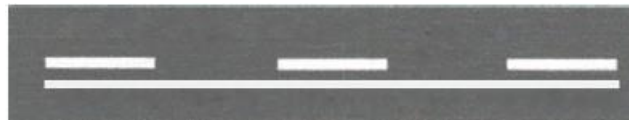
Gambar 3 Marka garis membujur putus-putus pada jalan 2 jalur, sesuai dengan peraturan Menteri Perhubungan

Sumber: (Menteri Perhubungan Republik Indonesia, 2018)

Gambar 3 menunjukkan panjang segmen 5 m, jarak antar garis 8 m, dan lebar garis 12 cm, Marka ini memainkan peran penting dalam menyediakan panduan visual bagi pengemudi dan membantu dalam mempertahankan aliran lalu lintas yang teratur, mengurangi risiko kecelakaan, dan meningkatkan keselamatan di jalan raya.

c. Marka garis membujur kombinasi penuh dan putus-putus

Gambar 4 menunjukkan marka garis membujur kombinasi penuh dan putus-putus sesuai dengan peraturan Menteri Perhubungan (Permenhub) tahun 2018. Marka jenis ini sering digunakan untuk memberikan petunjuk dan peraturan yang lebih kompleks bagi pengguna jalan. Misalnya, garis penuh menunjukkan bahwa pengemudi tidak diperbolehkan untuk melintasi atau berpindah jalur, sementara bagian garis putus-putus menandakan bahwa pengemudi diperbolehkan untuk melakukan hal tersebut di bawah kondisi tertentu.

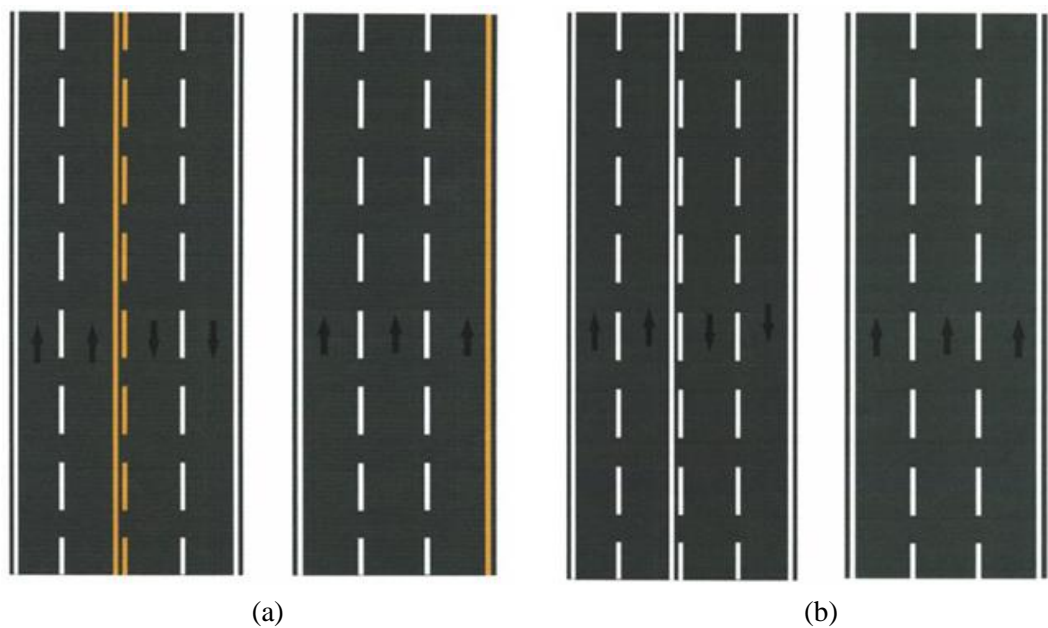


Gambar 4 Marka garis membujur kombinasi penuh dan putus-putus
Sumber: (Menteri Perhubungan Republik Indonesia, 2018)

Berdasarkan Gambar 4, marka garis membujur kombinasi penuh dan putus-putus biasanya ditempatkan di area di mana aturan lalu lintas berubah, seperti mendekati atau menjauhi persimpangan atau area di mana jalur keluar atau jalur tambahan dimulai. Penandaan ini berperan penting dalam keselamatan lalu lintas karena memberikan isyarat yang jelas kepada pengemudi mengenai tindakan yang diizinkan dan dilarang, sehingga membantu mencegah kecelakaan dan meningkatkan aliran lalu lintas yang aman.

2.1.3 Warna Marka Garis

Warna marka garis menampilkan dua variasi warna untuk marka garis membujur pada jalan. Perbedaan warna ini penting karena membantu pengemudi mengenali jenis jalan yang mereka gunakan, yang selanjutnya dapat mempengaruhi kecepatan berkendara dan tindakan yang harus diambil, seperti peraturan mendahului.



Gambar 5 Warna marka: (a) membujur jalan Nasional; (b) garis membujur jalan selain jalan Nasional

Sumber: (Menteri Perhubungan Republik Indonesia, 2018)

Gambar 5 bagian (a) menunjukkan marka jalan nasional yang memiliki warna khas, sedangkan bagian (b) menampilkan warna marka garis membujur yang digunakan untuk jalan-jalan selain jalan nasional. Penggunaan warna yang berbeda ini juga berperan dalam menyediakan konsistensi dan standarisasi dalam sistem penandaan jalan, yang memudahkan pengendara dari berbagai daerah untuk memahami dan mematuhi aturan lalu lintas saat berpindah antar jenis jalan.

2.2 Autonomous car

Autonomous car merupakan kendaraan canggih yang memiliki kemampuan untuk bergerak secara mandiri dari satu titik ke titik lain tanpa memerlukan bantuan

pengemudi. Dalam konteks ini, *autonomous car* dilengkapi dengan sistem *autopilot* yang memungkinkannya untuk menjalankan fungsi-fungsi operasional secara otomatis. Definisi yang lebih rinci menyebutkan bahwa *autonomous car* adalah kendaraan yang tidak hanya mampu bergerak secara mandiri, tetapi juga memiliki kemampuan untuk merasakan lingkungannya dan melakukan navigasi tanpa bergantung pada masukan manusia (Ondruš et al., 2020).

Pada tahun 2014, SAE *International (Society of Automotive Engineers)*, sebuah badan standarisasi otomotif, menerbitkan suatu sistem klasifikasi berdasarkan enam tingkatan yang berkisar dari tidak ada hingga sistem sepenuhnya otomatis. Sistem ini dijelaskan dalam standar J3016 dengan judul "*Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems*". Klasifikasi ini didasarkan pada tingkat intervensi dan kewaspadaan pengemudi yang diperlukan, bukan pada kemampuan kendaraan, meskipun keduanya sangat berkaitan. Pada 30 September 2016, SAE memperbarui standarnya untuk klasifikasi tingkat otomatisasi berkendara. Selain menetapkan tingkatan otomatisasi berkendara yang saling eksklusif, laporan teknis ini menetapkan terminologi standar untuk konsep-konsep yang terkait dengan kendaraan yang mengandung sistem otomatis. Standar SAE J3016 ini sejajar dengan tingkatan yang ditentukan oleh *Germany Federal Highway Research Institute (BASt)* dan sejauh ini sesuai dengan *National Highway Traffic Safety Administration (NHTSA)* (Ondruš et al., 2020). Berikut adalah keenam tingkatan tersebut:

Level 0 : Tanpa automasi, di mana pada tingkat ini sebagian besar mobil berada di tahap ini. pengemudi mengendalikan kemudi, gas dan rem, memantau keadaan sekitar serta menavigasi dan menentukan kapan harus menggunakan lampu sein, pindah jalur dan belok. Meskipun ada beberapa sistem peringatan (peringatan titik buta dan tabrakan).

Level 1 : Asisten pengemudi, di mana kendaraan pada tingkat ini dapat menangani kemudi atau gas dan rem, tetapi tidak dalam semua situasi, dan pengemudi harus siap untuk mengambil alih fungsi tersebut jika diminta oleh kendaraan. Artinya, pengemudi harus tetap menyadari apa yang dilakukan mobil dan siap untuk ikut campur jika diperlukan.

Level 2 : Automasi parsial, di mana pada tahap ini mobil menangani kemudi, gas, dan rem, tetapi segera membiarkan pengemudi mengambil alih jika mendeteksi objek atau peristiwa yang tidak direspon oleh mobil. Pada tiga tingkat pertama ini, pengemudi bertanggung jawab untuk memantau sekitar, lalu lintas, cuaca dan kondisi jalan.

Level 3 : Automasi bersyarat, di mana mobil memantau sekitar dan mengurus semua kemudi, gas, dan rem dalam lingkungan tertentu, seperti jalan bebas hambatan. Namun, pengemudi harus siap untuk campur tangan jika mobil memintanya.

Level 4 : Automasi tinggi, di mana mobil menangani kemudi, gas, dan rem, serta memantau sekitar dalam berbagai lingkungan, meskipun tidak semua, seperti cuaca ekstrem. Pengemudi hanya mengaktifkan sistem pengemudi otomatis saat kondisi aman untuk melakukannya. Setelah itu, pengemudi tidak diharuskan melakukan tindakan lebih lanjut.

Level 5 : Automasi sepenuhnya, di mana pengemudi hanya perlu menetapkan tujuan dan memulai mobil, selanjutnya mobil menangani semua tugas lainnya. Mobil dapat menuju ke tujuan apa pun dan membuat keputusan sendiri selama perjalanan.

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/Deceleration	Monitoring of Driving Environment	Fallback Performance of Dynamic Driving Task	System Capability (Driving Modes)
Human driver monitors the driving environment						
0	No Automation	the full-time performance by the <i>human driver</i> of all aspects of the <i>dynamic driving task</i> , even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	Driver Assistance	the <i>driving mode</i> -specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the <i>human driver</i> perform all remaining aspects of the <i>dynamic driving task</i>	Human driver and system	Human driver	Human driver	Some driving modes
2	Partial Automation	the <i>driving mode</i> -specific execution by one or more driver assistance systems of both steering and acceleration/deceleration using information about the driving environment and with the expectation that the <i>human driver</i> perform all remaining aspects of the <i>dynamic driving task</i>	System	Human driver	Human driver	Some driving modes
Automated driving system ("system") monitors the driving environment						
3	Conditional Automation	the <i>driving mode</i> -specific performance by an <i>automated driving system</i> of all aspects of the <i>dynamic driving task</i> with the expectation that the <i>human driver</i> will respond appropriately to a <i>request to intervene</i>	System	System	Human driver	Some driving modes
4	High Automation	the <i>driving mode</i> -specific performance by an <i>automated driving system</i> of all aspects of the <i>dynamic driving task</i> , even if a <i>human driver</i> does not respond appropriately to a <i>request to intervene</i>	System	System	System	Some driving modes
5	Full Automation	the full-time performance by an <i>automated driving system</i> of all aspects of the <i>dynamic driving task</i> under all roadway and environmental conditions that can be managed by a <i>human driver</i>	System	System	System	All driving modes

Gambar 6 SAE's levels of driving automation

Sumber: (Standard SAE J3016, 2016)

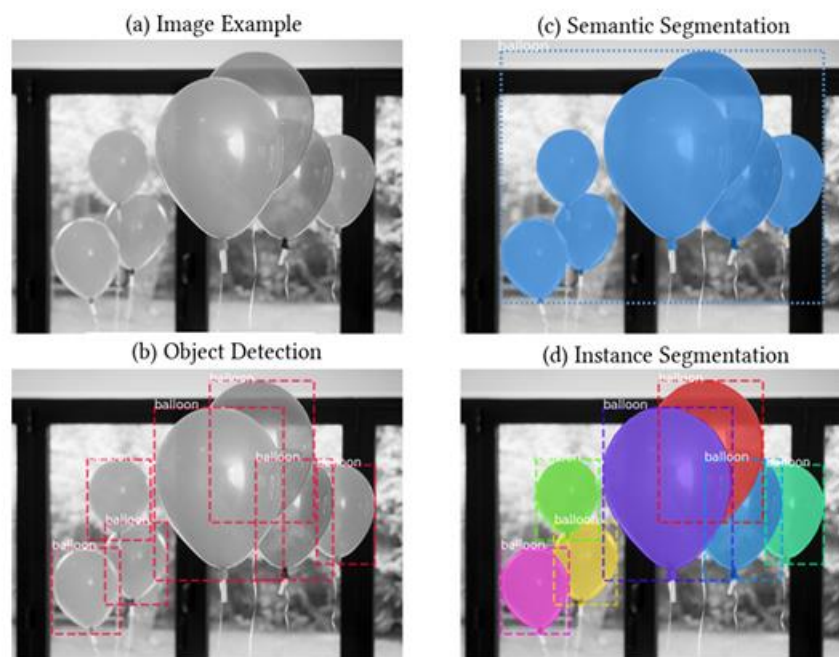
2.3 Visi Komputer

Visi komputer adalah sub-bidang *Deep learning* dan *Artificial Intelligence* dimana manusia mengajar komputer untuk melihat dan menafsirkan dunia di sekitar mereka. Visi komputer menjadi semakin penting dan efektif dalam beberapa tahun terakhir karena aplikasinya yang luas di berbagai bidang seperti pengawasan dan pemantauan cerdas (*smart surveillance*), kesehatan dan kedokteran, olahraga dan rekreasi, robotika, *drone*, mobil *self-driving*, dan lain sebagainya. Tugas pengenalan visual, seperti klasifikasi gambar, pelokalan, dan deteksi, adalah inti dari pemanfaatan aplikasi ini yang mana telah menghasilkan kinerja yang mumpuni dalam tugas dan sistem pengenalan visual (Marpaung et al., n.d., 2022).

Saat ini, visi komputer adalah sebuah divisi *artificial intelligence* (AI) yang menangani pengajaran komputer bagaimana 'melihat' gambar dan objek 2D dan 3D. Didukung oleh *deep learning*, visi komputer mahir dalam mengidentifikasi dan memproses data visual dalam jumlah besar dengan cepat dan akurat dan membuat keputusan atau rekomendasi berdasarkan informasi tersebut (Marpaung et al., n.d., 2022).

Penerapan visi komputer salah satunya yaitu pada *autonomous car* mencakup pada deteksi dan identifikasi objek. Teknologi *autonomous car* memungkinkan kendaraan beroperasi secara mandiri dengan memperhatikan lingkungan dan memberikan respons yang responsif menggunakan kamera ataupun LiDAR (*Light Detection and Ranging*) (Grigorescu et al., 2020)

Autonomous car memanfaatkan visi komputer untuk meningkatkan efisiensi dan keamanan dalam penggunaan sehari-hari. Teknologi ini dapat digunakan untuk membuat peta tiga dimensi, mendeteksi dan mengklasifikasikan objek, serta mengumpulkan data yang membantu *autonomous car* menjadi sadar akan lingkungan sekitarnya dan bertindak sesuai dengan kondisi yang ada. Visi komputer adalah bagian yang sangat penting dalam teknologi, dan ini terkait dengan cara komputer memproses gambar. Fungsinya mirip dengan membuat gambar digital dapat dimengerti oleh mesin. Dalam konteks *autonomous car*, visi komputer berperan dalam mengolah gambar, *pixel*, dan menganalisis fitur. Ini menjadi alat kunci untuk membuat proses menjadi otomatis dan otonom dalam bidang *autonomous car* (Gajjar & Sanyal, 2023).



Gambar 7 Perbedaan tugas visi komputer
Sumber: (Abdulla, 2018)

Dalam pengolahan citra pada gambar 7, terdapat berbagai pendekatan untuk memahami gambar. Klasifikasi mengidentifikasi label keseluruhan gambar, seperti "ada sebuah balon di dalam gambar". Sementara itu, *semantic segmenation* memetakan setiap *pixel* ke kelas yang berbeda, memberikan informasi detail tentang distribusi *pixel* yang berkaitan dengan balon. Pada tingkat yang lebih lanjut, *object detection* memberikan lokasi relatif dan jumlah objek dalam gambar, serta kotak pembatas di sekitar setiap objek. Kemudian, *instance segmentation* memperinci lebih lanjut dengan memisahkan setiap objek dalam gambar, memberikan informasi tentang *pixel* yang termasuk ke dalam setiap objek secara individual. Ini memberikan pemahaman yang lebih mendalam tentang struktur objek dan memungkinkan identifikasi objek yang tumpang tindih satu sama lain (Abdulla, 2018).

2.4 Semantic Segmentation

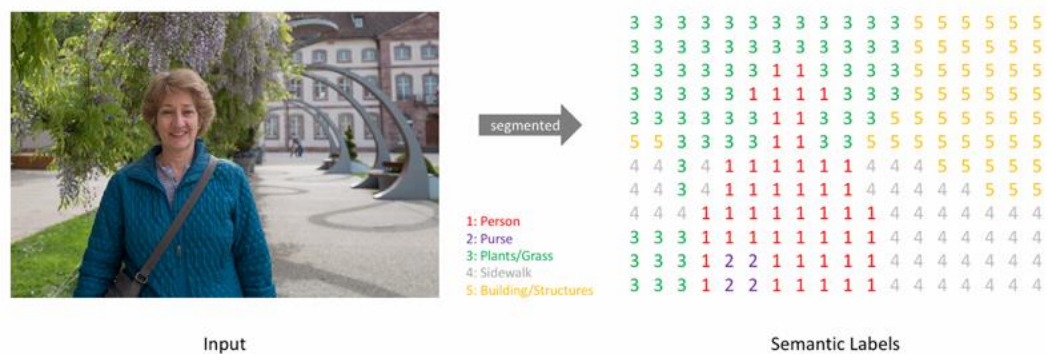
Semantic segmentation adalah sebuah pekerjaan penting dalam bidang visi komputer yang bertujuan untuk memberikan label yang tepat pada setiap *pixel* dalam gambar berdasarkan isi semantiknya. Tugas ini sering disebut sebagai

pelabelan padat karena melibatkan klasifikasi gambar pada tingkat *pixel* (Guo et al., 2023). *Semantic Segmentation* dikenal sebagai proses pemisahan latar depan karena fokus pada penonjolan subjek gambar, bukan pada elemen latar belakang. Dalam *semantic segmentation*, setiap *pixel* diidentifikasi dan diklasifikasikan berdasarkan kelas objeknya, seperti jalan, mobil, pejalan kaki, bangunan, dan lain sebagainya (Marpaung et al., n.d., 2022).

Salah satu contoh awal *semantic segmentation* dapat ditelusuri kembali ke pengembangan *Berkeley Segmentation Dataset and Benchmark* (BSDS) di awal tahun 2000-an, yang bertujuan untuk mengevaluasi algoritma *image segmentation*. Selama bertahun-tahun, kemajuan dalam bidang *deep learning*, khususnya dengan diperkenalkannya *Fully Convolutional Networks* (FCNs) telah meningkatkan akurasi dan efisiensi tugas *semantic segmentation* secara signifikan (Chib & Singh, 2023).

Perkembangan terkini dalam *semantic segmentation* mencakup integrasi *attention mechanisms*, penggabungan fitur multi-skala, dan *generative adversarial networks* (GANs) untuk meningkatkan kinerja segmentasi. Kemajuan ini telah menghasilkan penerapan di berbagai bidang seperti *autonomous car*, analisis citra medis, dan penginderaan jauh, di mana pemahaman semantik gambar yang tepat sangat penting untuk proses pengambilan keputusan (Chib & Singh, 2023).

Tujuan dari *semantic segmentation* dalam pengolahan citra komputer adalah untuk memberikan pemahaman yang lebih dalam tentang konten gambar dengan melakukan klasifikasi *pixel-pixel* dalam gambar berdasarkan label semantiknya. Teknik ini membantu dalam memisahkan objek-objek yang berbeda dalam gambar dan memungkinkan pemrosesan yang lebih akurat dan kontekstual terhadap informasi visual yang ada (Guo et al., 2023).



Gambar 8 Semantic Segmentation secara Holistic

Sumber: (Singh, 2022)

Image segmentation dapat melibatkan pemisahan latar depan dari latar belakang, atau pengelompokan wilayah *pixel* (*superpixel*) yang termasuk dalam kelas yang sama. Manfaat utama dari *image segmentation* adalah pemahaman situasi. *Semantic Segmentation* pada gambar 8 mampu menangani dua permasalahan utama, yakni identifikasi objek dan penentuan lokasi mereka. Dengan teknik ini, *Semantic Segmentation* dapat mengidentifikasi dan mengklasifikasikan objek seperti orang, tas, tanaman, trotoar, dan bangunan dalam sebuah gambar. Tidak hanya itu, *Semantic Segmentation* juga dapat memberikan informasi tentang lokasi relatif dari setiap objek yang teridentifikasi dalam gambar tersebut. Sehingga, secara efektif, teknik *Semantic Segmentation* memungkinkan untuk mengklasifikasikan objek-objek yang berbeda dan menentukan posisi relatif mereka dalam sebuah gambar.

Dengan penggunaan arsitektur *deep learning* seperti *convolutional neural networks* (CNN) yang telah menunjukkan kinerja yang luar biasa dalam menghasilkan fitur-fitur hierarkis dan semantik tingkat tinggi dari gambar. Teknologi *deep learning* ini telah memainkan peran penting dalam meningkatkan akurasi dan efisiensi dari metode *semantic segmentation*, serta menarik minat dari berbagai bidang teknis dan penelitian yang membutuhkan kemampuan visi komputer (Guo et al., 2023).

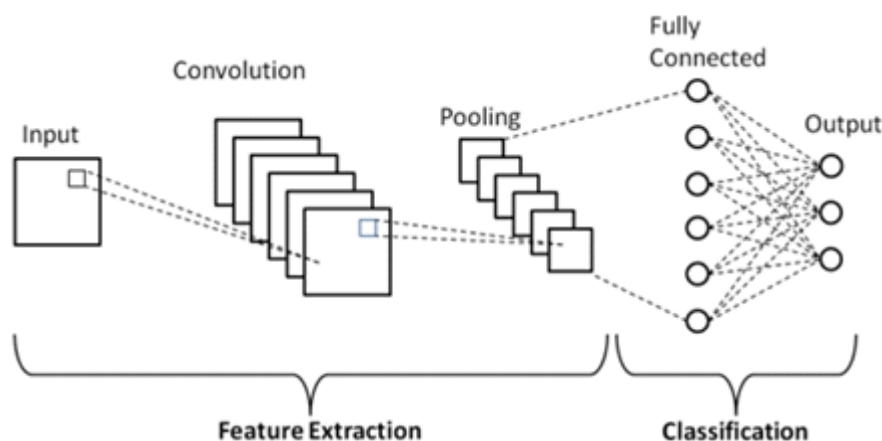
2.5 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan jenis jaringan saraf yang menggunakan operasi konvolusi, pengumpulan (*pooling*), serta berbagai fungsi aktivasi untuk mengekstrak fitur dari data input. CNN telah menjadi populer dalam

berbagai aplikasi seperti klasifikasi gambar, deteksi objek, dan segmentasi karena kemampuannya dalam memahami dan merepresentasikan data yang kompleks. Seiring berjalannya waktu, arsitektur CNN terus berkembang dengan peningkatan dalam skalabilitas, kompleksitas, dan kinerja, yang menghasilkan berbagai model CNN yang disesuaikan untuk berbagai tugas dan aplikasi (Patel & Patel, 2020).

Pada dasarnya, *Convolutional Neural Network* (CNN) adalah salah satu jenis dari *Neural Network* yang merupakan arsitektur *deep learning* yang terinspirasi dari cara kerja otak manusia. CNN pertama kali diperkenalkan sekitar tahun 1998 dan telah mengalami perkembangan yang pesat sejak itu. CNN merupakan modifikasi dari jaringan saraf tradisional, di mana tidak semua *neuron* terhubung secara penuh satu sama lain. Di CNN, koneksi antara *neuron* jarang terjadi secara penuh, dan setiap lapisan dalam CNN dapat berperilaku dengan cara yang berbeda (Marpaung et al., n.d., 2022).

CNN terbukti sebagai algoritma pembelajaran terbaik untuk memahami informasi yang tersedia ke dalam citra dan dianggap sebagai model ideal untuk berbagai tugas terkait citra seperti segmentasi, klasifikasi, penandaan, deteksi dan lainnya. Sama halnya seperti otak manusia, CNN terdiri dari banyak sel komputasi yang disebut sebagai '*neuron*'. Sel-sel ini melakukan operasi sederhana dan berinteraksi satu sama lain untuk membuat keputusan (Patel & Patel, 2020).



Gambar 9 Lapisan arsitektur dalam CNN

Sumber: (Marpaung et al., n.d., 2022)

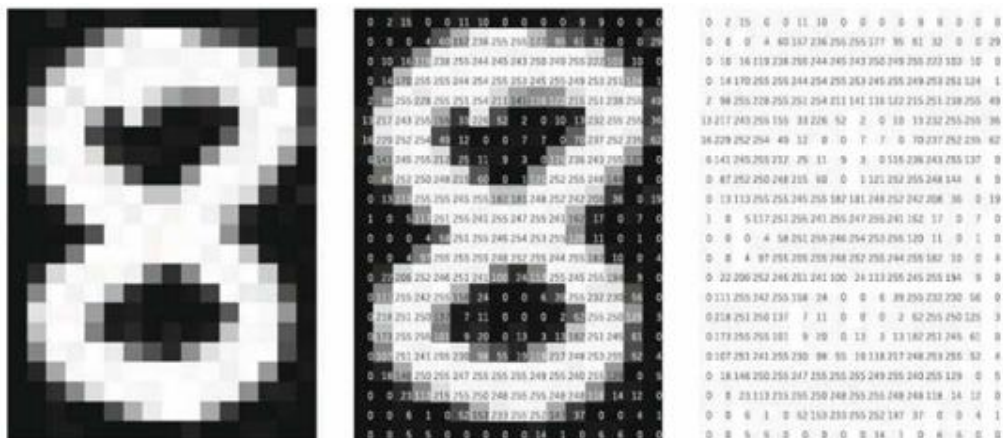
Nama CNN diberikan berdasarkan jenis lapisan tersembunyi yang dikandungnya, terdiri dari beberapa blok bangunan dasar. Ada lapisan *convolutional*, lapisan *pooling*, lapisan *fully connected*, dan lapisan normalisasi

lapisan tersembunyi yang digunakan oleh CNN. Pada Gambar 9, klasifikasi gambar umum ditampilkan, lapis demi lapis menggunakan arsitektur CNN (Marpaung et al., n.d., 2022).

A. Convolutional Layer

Convolutional layer adalah lapisan pertama CNN yang mengekstrak fitur dari gambar masukan dengan menerapkan filter atau kernel ke data masukan. Lapisan ini membantu menjaga hubungan spasial antar *pixel* pada gambar masukan dan mempelajari fitur gambar melalui operasi konvolusi (Patel & Patel, 2020).

Fungsi utama dari lapisan konvolusi adalah untuk mendeteksi pola spasial dalam data masukan, seperti tepi, tekstur, atau bentuk, melalui operasi konvolusi. Dengan mempelajari fitur hierarki dari pola tingkat rendah hingga tingkat tinggi, lapisan konvolusional memainkan peran penting dalam menangkap struktur kompleks dalam gambar. Ekstraksi fitur hierarki ini memungkinkan CNN mengenali objek dan pola dalam gambar dengan tingkat abstraksi yang semakin meningkat (Yamashita et al., 2018).

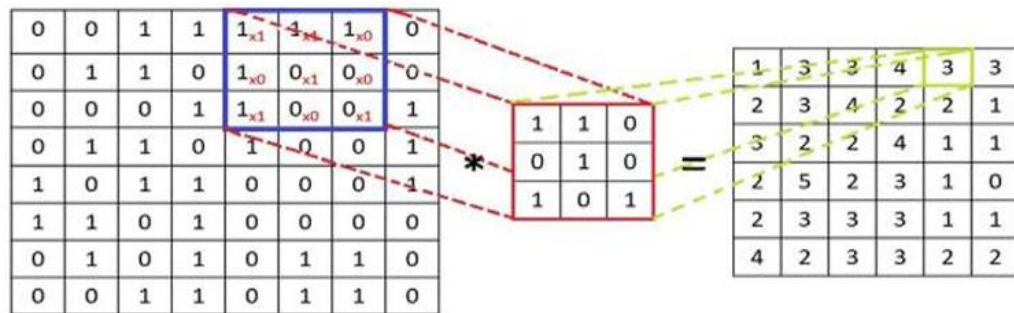


Gambar 10 Nilai gambar dalam bentuk matrix

Sumber: (Yamashita et al., 2018)

Lapisan konvolusional sangat penting dalam tugas pemrosesan gambar karena membantu mengidentifikasi dan mengekstrak fitur-fitur penting dari gambar, yang kemudian digunakan untuk tugas-tugas seperti klasifikasi

gambar, deteksi objek, dan segmentasi. Melalui operasi konvolusi, lapisan menggabungkan data masukan dengan kernel yang dapat dipelajari untuk membuat peta fitur yang menyoroti karakteristik penting dari gambar masukan. Proses ekstraksi fitur ini sangat penting bagi CNN untuk memahami dan menafsirkan informasi visual secara efektif (Yamashita et al., 2018).



Gambar 11 Convolutional Layer

Sumber: (Purwono et al., 2022).

Lapisan konvolusional menggunakan filter kernel yang lebih kecil daripada gambar masukan untuk menghitung konvolusi, mengekstraksi fitur-fitur mendasar. Filter kernel memiliki ukuran dimensi yang tetap dan nilai parametereternya konstan. Kernel filter berukuran $f \times f \times 2$, di mana f adalah 3, 5, 7, dan seterusnya. Filter harus lebih kecil daripada gambar masukan, dan kernel filter bergerak melintasi gambar masukan langkah demi langkah, menghasilkan peta aktivasi 2D. CNN mempelajari fitur visual dari gambar tersebut. Persamaan umum lapisan konvolusional dapat dinyatakan seperti pada (1) dan (2). Gambar 11 menunjukkan ilustrasi sederhana proses komputasi di CNN yang menghasilkan peta aktivasi (Purwono et al., 2022).

$$Activation\ map\ (p, q) = Input * Filter \quad (1)$$

$$Am\ (p, q) = \sum_{y=0}^{Columns} \left(\sum_{x=0}^{row} (x - p, y - q) \times (x, y) \right) \quad (2)$$

Keterangan:

Input = *matrix input*

Filter = *matrix filter* atau *kernel* yang ditetapkan pada input

x dan y = indeks yang digunakan untuk iterasi melalui elemen dari *filter*

p dan q = *offset* dari posisi saat ini di input terhadap posisi filter

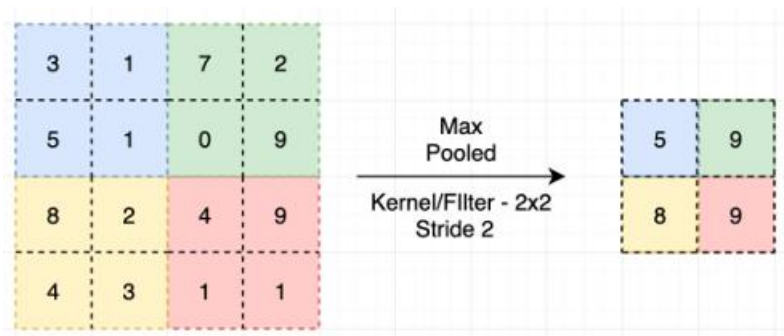
Activation map = *output* dari *convolusi*, yang sering disebut *feature map* atau *activation map*, yang berisi informasi tentang fitur-fitur tertentu yang telah dideteksi filter di posisi tertentu

B. Pooling

Pooling layer dalam CNN adalah komponen yang melakukan operasi *downsampling* untuk mengurangi dimensi spasial dari peta fitur, memperkenalkan invariansi translasi terhadap pergeseran kecil, dan mengurangi jumlah parameter yang dapat dipelajari selanjutnya. Dengan mengurangi dimensi spasial dari peta fitur, *pooling* meningkatkan efisiensi komputasi dalam CNN dan membantu dalam pembentukan fitur-fitur hierarkis dengan representasi data (Yamashita et al., 2018).

Jenis-jenis *pooling* yang umum digunakan dalam CNN adalah *max pooling* dan *average pooling*. *Max pooling* mengekstrak nilai maksimum dari setiap *patch* pada *feature map*, sementara *average pooling* mengambil nilai rata-rata dari setiap *patch*. *Max pooling* membantu dalam mempertahankan fitur penting dalam data, sementara *average pooling* dapat mengurangi *overfitting* dan mempercepat proses pembelajaran (Yamashita et al., 2018).

Max pooling secara khusus sangat efektif dalam mengurangi dimensi data sambil mempertahankan fitur penting. Dengan mengambil hanya nilai maksimum dari setiap wilayah yang dipilih, teknik ini memastikan bahwa CNN dapat fokus pada aspek paling signifikan dari input tanpa terbebani oleh setiap detail *pixel*. Ini tidak hanya meningkatkan invariansi terhadap pergeseran citra, tetapi juga mengoptimalkan efisiensi komputasi, yang penting dalam proses pelatihan dan inferensi yang lebih cepat (Patel & Patel, 2020).

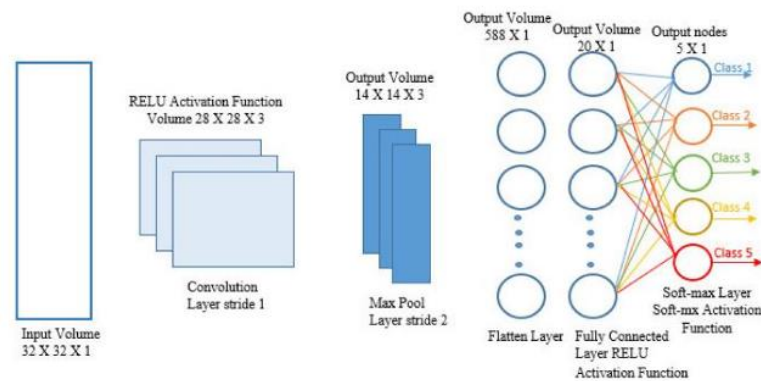


Gambar 12 Contoh *max pooling* pada CNN
 Sumber: (Marpaung et al., n.d., 2022)

Lapisan *fully connected* di CNN berada di ujung arsitektur dan menggabungkan fitur yang diekstrak oleh lapisan konvolusi dan *pooling* untuk klasifikasi. Setiap *neuron* di lapisan ini terhubung ke semua *neuron* sebelumnya, memungkinkan pengenalan pola kompleks. Gambar 12 mengilustrasikan *max pooling* yang efisien mengurangi ukuran peta fitur sambil menonjolkan fitur kunci. Ini merupakan bagian penting dari CNN untuk klasifikasi gambar yang efektif (Marpaung et al., n.d., 2022).

C. *Fully Connected Layer*

Fully Connected Layer dalam CNN adalah lapisan yang umumnya disebut sebagai lapisan *output* konvolusi. Lapisan ini mirip dengan *feedforward neural network* dan biasanya ditemukan di bagian bawah jaringan. *Fully Connected Layer* menerima input dari lapisan *pooling* terakhir atau lapisan *output* konvolusi, yang kemudian diluraskan sebelum dikirim ke lapisan berikutnya. *Output* dari lapisan sebelumnya diubah menjadi vektor dengan nilai-nilai yang merata, sehingga memungkinkan studi kombinasi *non-linear* tingkat tinggi dari fitur yang direpresentasikan oleh lapisan *output* konvolusi (Purwono et al., 2022).

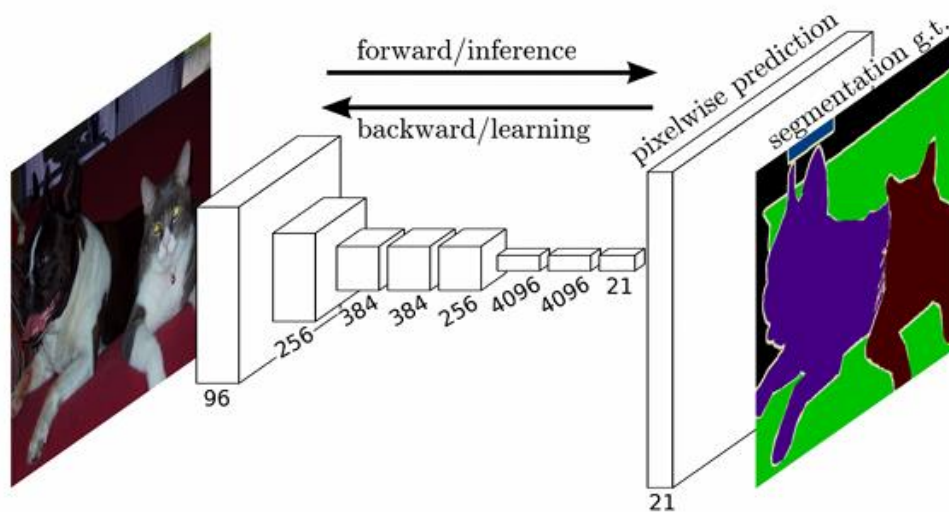


Gambar 13 *Fully Connected Layer*
 Sumber: (Purwono et al., 2022)

Perbedaan antara *Fully Connected Layer* pada gambar 13 dengan lapisan konvolusi dan *pooling* terletak pada cara informasi diproses. Lapisan konvolusi bertanggung jawab untuk mengekstrak fitur dari input, sedangkan lapisan *pooling* mengurangi dimensi data dengan melakukan *down-sampling*. *Fully Connected Layer* kemudian menggunakan fitur-fitur yang diekstrak sebelumnya untuk melakukan klasifikasi dan prediksi. Lapisan ini menghubungkan semua aktivasi *neuron* dari lapisan sebelumnya ke lapisan berikutnya, memungkinkan jaringan untuk mempelajari hubungan yang lebih kompleks antara fitur-fitur tersebut. Dalam *image segmentation* menggunakan *fully connected layer* untuk memisahkan gambar menjadi bagian-bagian yang berbeda berdasarkan fitur-fitur yang ada (Purwono et al., 2022).

2.6 *Fully Convolutional Network For Semantic Segmentation*

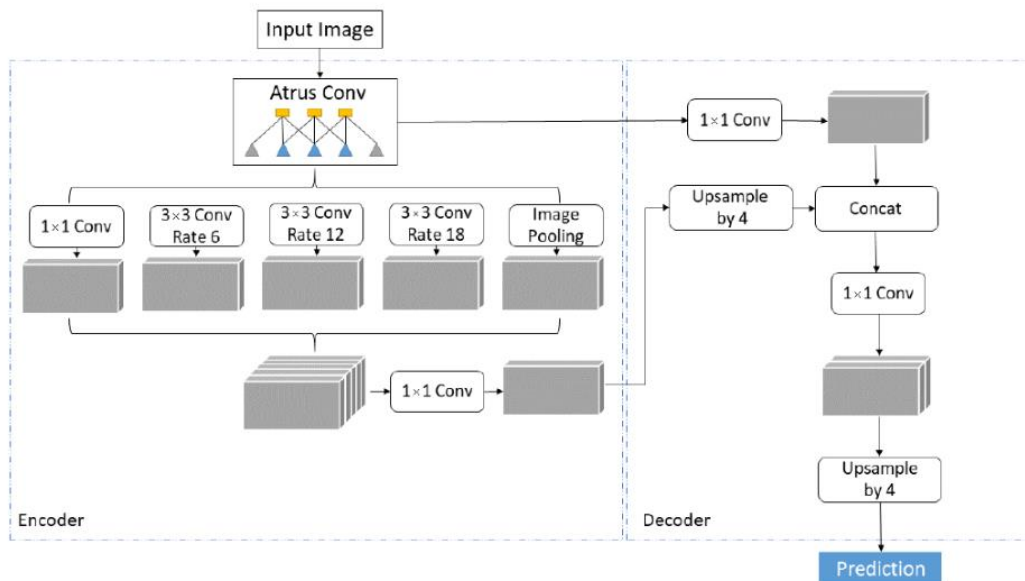
Fully Convolutional Network (FCN) merupakan model visual yang menggunakan jaringan konvolusional untuk *semantic segmentation*. FCN dilatih secara *end-to-end*, artinya dari *pixel* ke *pixel*, untuk menghasilkan hasil yang canggih dalam tugas *semantic segmentation*. FCN dirancang untuk dapat menerima input dengan ukuran apapun dan mengeluarkan *output* dengan ukuran yang sesuai secara efisien, sehingga memungkinkan prediksi yang padat secara spasial. Ide utama di balik FCN adalah membangun jaringan yang mampu melakukan prediksi per *pixel* untuk tugas seperti *semantic segmentation* (Long et al., 2014).



Gambar 14 Arsitektur FCN
 Sumber: (B. Li et al., 2018)

FCN pada gambar 14 menggunakan kombinasi lapisan konvolusi dan lapisan *upsampling* dalam jaringannya untuk memfasilitasi prediksi dan pembelajaran *pixel* dengan pengumpulan subsampel. Dengan menyatukan hierarki fitur dari lapisan-lapisan tersebut dan menyempurnakan presisi spasial *output*, FCN mampu menghasilkan segmentasi yang detail. Penggunaan lapisan *upsampling* memungkinkan FCN untuk memprediksi *output* yang padat dari masukan dengan ukuran apapun, memungkinkan pembelajaran dan inferensi pada seluruh gambar secara bersamaan melalui proses *feedforward* yang padat serta *backpropagation* (Long et al., 2014).

2.7 DeepLabv3+



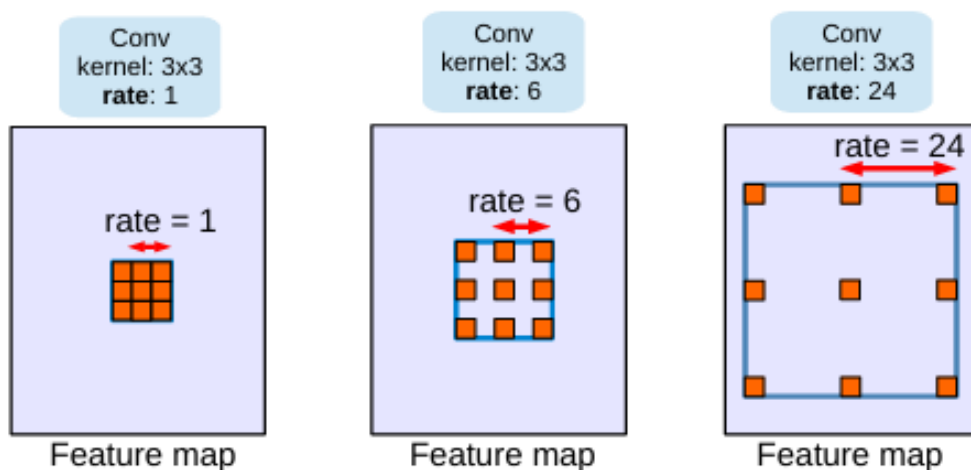
Gambar 15 Arsitektur *Encoder-Decoder* mendetail dari *DeepLabv3+* untuk *semantic segmentation*

Sumber: (Štifanić et al., 2021)

Langkah awal dalam proses arsitektur *DeepLabv3+* pada gambar 15 dimulai dengan gambar input yang akan melalui lapisan *encoder* yang dirancang untuk mengolah berbagai detail semantik dari gambar tersebut. Proses *encoder* dalam arsitektur *DeepLabv3+* dimulai dengan penggunaan *Atrous Spatial Pyramid Pooling* (ASPP) untuk menangkap informasi semantik yang luas dan multi-skala dari citra input. *Atrous convolution* digunakan dalam modul *encoder* untuk memungkinkan penggunaan filter dengan jarak yang dapat disesuaikan, sehingga mampu menangkap konteks yang luas tanpa mengurangi resolusi citra. ASPP dalam *DeepLabv3+* melibatkan beberapa *atrous convolution* dengan tingkat dilasi (6, 12, 18) yang berbeda untuk memproses citra pada berbagai skala, memungkinkan model untuk menangkap konteks yang luas dan detail. Selain itu, modul *encoder* ini menggunakan struktur *encoder-decoder*, di mana fitur-fitur yang dihasilkan diproses dengan *atrous convolution* untuk mengurangi resolusi, memungkinkan model untuk menangkap konteks yang luas tanpa mengurangi resolusi citra secara signifikan. Penggunaan *depthwise separable convolution*, yang memisahkan proses konvolusi dalam dua tahap, membantu mengurangi kompleksitas komputasi tanpa

mengurangi kinerja. Setelah fitur-fitur yang kaya informasi semantik dihasilkan oleh modul encoder, proses berlanjut ke modul *decoder* (Chen et al., 2018).

Modul *decoder* dalam arsitektur *DeepLabv3+* bertujuan untuk memulihkan informasi spasial yang lebih rinci dari fitur-fitur yang telah diproses oleh modul *encoder*. Proses ini dimulai dengan *bilinear upsampling* fitur-fitur dari modul *encoder* dengan faktor 4 untuk meningkatkan resolusi fitur-fitur tersebut. Fitur-fitur ini kemudian digabungkan dengan fitur-fitur *low-level* dari jaringan *backbone* yang mengandung informasi spasial yang lebih detail. Setelah penggabungan, dilakukan pengurangan jumlah *channel* menggunakan konvolusi 1×1 untuk memastikan fitur-fitur *encoder* tetap dominan dan mengurangi kompleksitas. Fitur-fitur yang digabungkan kemudian diproses dengan beberapa konvolusi 3×3 untuk memperhalus dan memperbaiki fitur-fitur tersebut. Akhirnya, dilakukan *bilinear upsampling* kedua dengan faktor 4 untuk memulihkan resolusi yang lebih rinci, menghasilkan output segmentasi citra semantik yang lebih akurat dengan batasan objek yang jelas. Dengan demikian, arsitektur *DeepLabv3+* efektif dalam menangkap informasi semantik yang luas dan multi-skala serta memulihkan informasi spasial yang lebih rinci, menghasilkan fitur-fitur yang kaya informasi untuk segmentasi citra semantik yang akurat (Chen et al., 2018).



Gambar 16 *Atrous convolution* dengan kernel 3×3 dan rate yang berbeda
Sumber: (Chen et al., 2017b)

Atrous convolution pada gambar 16 adalah sebuah konsep dalam jaringan konvolusi yang memungkinkan pengguna untuk mengontrol resolusi fitur yang

diekstrak tanpa menambah jumlah parameter. Dalam *atrous convolution*, lubang atau rate diterapkan pada filter konvolusi untuk memperluas bidang pandang (*receptive field*) tanpa menambah parameter (Chen et al., 2018). Dengan menyesuaikan laju *atrous*, yang menentukan jarak pengambilan sampel masukan, *atrous convolution* dapat mengontrol resolusi dan ukuran bidang pandang filter. Hal ini dicapai dengan menggabungkan masukan dengan filter yang di-*upsampled* yang memiliki angka nol yang disisipkan di antara nilai-nilai yang berurutan berdasarkan laju *atrous*. *Atrous convolution* memungkinkan modifikasi adaptif bidang pandang filter dengan mengubah nilai laju, sehingga konteks yang lebih luas dapat ditangkap tanpa menambah kompleksitas pada model (Chen et al., 2017b). Untuk setiap posisi i pada *output* y dan filter w , *atrous convolution* diterapkan pada peta fitur input x dalam domain dua dimensi seperti pada rumus 3 berikut.

$$y [i] = \sum_k x [i + r \cdot k]w[k] \quad (3)$$

Keterangan:

y = *Output* dari operasi konvolusi

i = Indeks posisi pada *output*

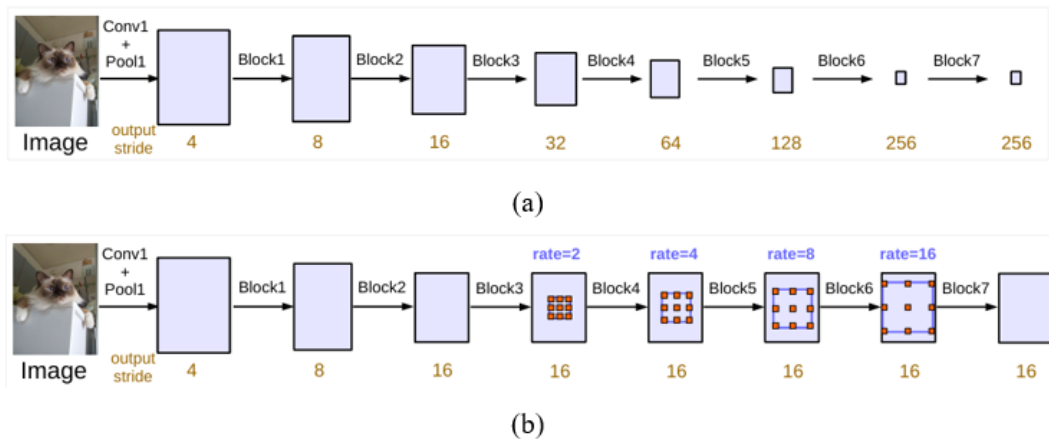
x = Sinyal input atau array yang akan dikonvolusi

r = Faktor dilasi dalam *atrous convolution*, menentukan jarak antara elemen-elemen dalam kernel konvolusi

k = Indeks yang digunakan dalam penjumlahan, mewakili elemen-elemen dari kernel konvolusi

w = Kernel konvolusi atau filter yang digunakan dalam operasi konvolusi

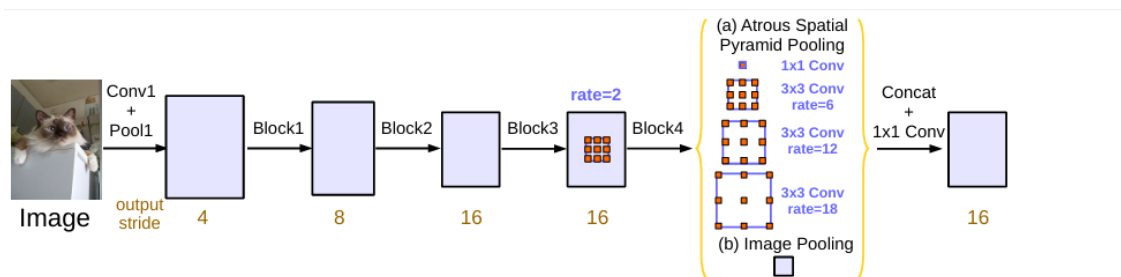
Dalam konteks *image segmentation* atau pengenalan pola, *atrous convolution* bermanfaat karena memungkinkan jaringan menangkap informasi kontekstual pada berbagai skala. Dengan menggunakan *atrous convolution*, jaringan dapat memperluas bidang pandangnya tanpa menambah jumlah parameter, memungkinkan ekstraksi fitur padat dan meningkatkan kinerja dalam tugas-tugas seperti *image segmentation* atau pengenalan pola (Chen et al., 2017).



Gambar 17 Konvolusi: (a) tanpa *atrous convolution*; (b) dengan *atrous convolution*
 Sumber: (Chen et al., 2017)

Perbedaan antara *traditional convolution* dan *atrous convolution* pada gambar 17 dapat diilustrasikan dalam cara filter diterapkan pada input. Dalam *traditional convolution*, filter diterapkan pada input tanpa meloncati *pixel*, artinya filter mengambil informasi langsung dari area yang berdekatan. Dengan demikian, bidang pandangnya terbatas pada area sekitar filter, yang mungkin menghasilkan keterbatasan dalam menangkap informasi kontekstual yang luas (Chen et al., 2017).

Di sisi lain, *atrous convolution* memperluas bidang pandang tanpa menambah parameter dengan memungkinkan filter meloncati *pixel* berdasarkan *atrous rate*. Ini berarti filter diterapkan pada input dengan meloncati beberapa *pixel*, memungkinkan filter untuk mengakses informasi dari jarak yang lebih jauh. Akibatnya, bidang pandangnya menjadi lebih luas dan lebih komprehensif, yang dapat meningkatkan kemampuan jaringan untuk menangkap konteks yang penting dalam data gambar (Chen et al., 2017).



Gambar 18 *Atrous Spatial Pyramid Pooling (ASPP)*
 Sumber: (Chen et al., 2017)

Atrous Spatial Pyramid Pooling (ASPP) adalah sebuah metode yang menggabungkan beberapa *atrous convolution* dengan tingkat *atrous* yang berbeda pada peta fitur. ASPP terinspirasi dari keberhasilan *spatial pyramid pooling* dalam mengambil sampel fitur pada skala yang berbeda untuk mengklasifikasikan wilayah dengan skala yang beragam. ASPP efektif dalam menangkap informasi multi-skala dengan tingkat *atrous* yang berbeda. Namun, semakin besar nilai sampling rate, jumlah bobot filter yang valid menjadi lebih sedikit. Hal ini dapat mengakibatkan filter degenerasi menjadi filter sederhana 1x1 saat nilai rate mendekati ukuran peta fitur, karena hanya bobot filter tengah yang efektif (Chen et al., 2017).

Pada penelitian yang dilakukan Chen et al menggunakan *pascal voc 2012 dataset* dan *cityscapes dataset* membandingkan kinerja *DeepLabv3+* untuk *image segmentation*. Hasilnya *DeepLabv3+* menempati akurasi tertinggi dengan menggunakan matriks IoU yang dapat dilihat pada Tabel 1.

Tabel 1. Hasil Segmentasi (IoU) pada *PASCAL VOC 2012 dataset*

Method	mIOU
Deep Layer Cascade (LC)	82.7
TuSimple	83.1
Large Kernel Matters	83.6
Multipath-ResNet	84.2
ResNet-38 MS COCO	84.9
PSPNet	85.4
IDW-CNN	86.3
CASIA IVA SDN	86.6
DIS	86.8
DeepLabv3	85.7
DeepLabv3-JFT	86.9
DeepLabv3+ (Xception)	87.8
DeepLabv3+ (Xception-JFT)	89.0

Sumber: (Chen et al., 2018b)

Sedangkan untuk *cityscapes dataset*, hasil segmentasi menggunakan *DeepLabv3+* menunjukkan kinerja yang lebih baik daripada metode lainnya, sebagaimana terlihat pada Tabel 2.

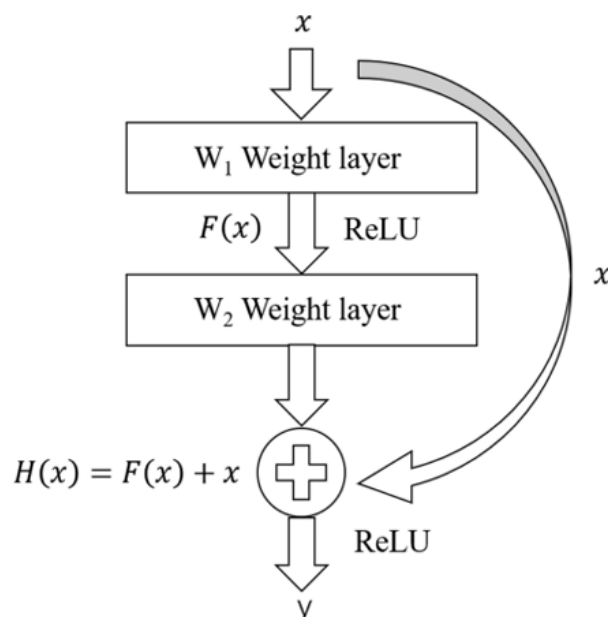
Tabel 2. Hasil Segmentasi (IoU) pada *cityscapes dataset*

Method	mIOU
ResNet-38	80.6
PSPNet	81.2
Mapillary	82.0
DeepLabv3	81.3
DeepLabv3+	82.1

Sumber: (Chen et al., 2018b)

2.8 ResNet 18 Backbone

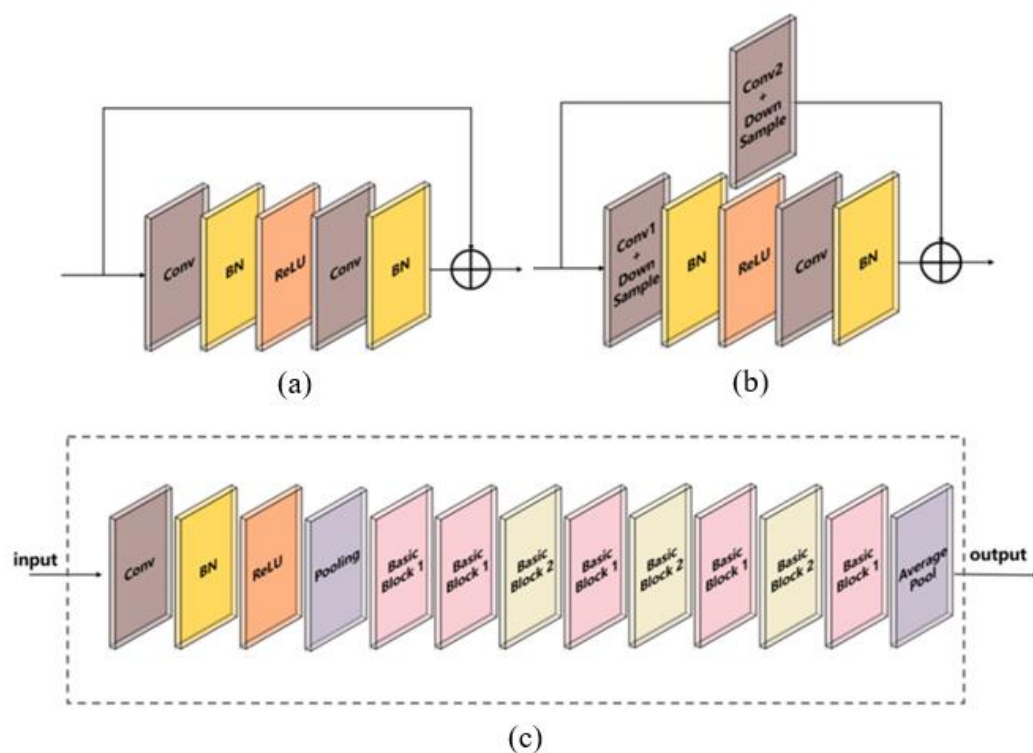
ResNet merupakan jaringan saraf konvolusional yang umum digunakan dan terdiri dari susunan blok-blok residual. Blok residual diperlihatkan pada Gambar 19, di mana x dan y adalah input dan output dari blok tersebut. W_1 dan W_2 mewakili bobot lapisan pertama dan kedua. Panah melengkung menunjukkan koneksi shortcut. $F(x)$ adalah output dari lapisan pertama setelah mengalami transformasi linier dan aktivasi. Setelah melalui transformasi linier oleh lapisan bobot W_2 , $F(x)$ dan input asli x ditambahkan untuk memperoleh $H(x)$, yang kemudian diaktifkan oleh fungsi ReLU untuk menghasilkan output y . Berkat adanya blok-blok residual ini, ResNet dapat mengoptimalkan lapisan jaringan dan mengurangi redundansi (B. Li & Gu, 2023).



Gambar 19 *Residual Block*

Sumber: (B. Li & Gu, 2023)

Sesuai yang ditunjukkan pada Gambar 20c, backbone ResNet18 terdiri dari enam tahap, termasuk *Conv 1*, *Layer 1*, *Layer 2*, *Layer 3*, *Layer 4*, dan *Pool*. *Conv 1* terdiri dari lapisan konvolusional, lapisan *batch normalization*, lapisan ReLU, dan lapisan *pooling*. Ukuran kernel, ukuran *stride*, dan ukuran *padding* dari lapisan konvolusional masing-masing adalah 7x7, 2, dan 3. Ukuran kernel, ukuran *stride*, dan ukuran *padding* dari lapisan *pooling* masing-masing adalah 3x3, 2, dan 1. *Layer 1* terdiri dari dua *Basic Block 1*. *Layer 2*, *Layer 3*, dan *Layer 4* terdiri dari *Basic Block 1* dan *Basic Block 2*. Kedua *Basic Block* ditunjukkan pada Gambar 20a dan 20b (B. Li & Gu, 2023).



Gambar 20 Skema diagram ResNet-18 backbone: (a) *Basic Block 1*; (b) *Basic Block 2*; (c) Resnet 18 Backbone

Sumber: (B. Li & Gu, 2023)

Basic Block 1 terdiri dari dua lapisan konvolusi, di mana ukuran *kernel*, ukuran *stride*, dan ukuran *padding* dari kedua lapisan konvolusi ini sama, yaitu 3x3, 1, dan 1. *Basic Block 2* memiliki tiga lapisan konvolusi, dengan dua lapisan pertama berfungsi sebagai lapisan *down-sampling*. Ukuran *kernel*, ukuran *stride*, dan ukuran *padding* dari lapisan konvolusi *down-sampling* pertama adalah 3x3, 2, dan 1. Sedangkan ukuran *kernel*, ukuran *stride*, dan ukuran *padding* dari lapisan konvolusi *down-sampling* kedua adalah 1x1, 2, dan 1. Untuk lapisan konvolusi ketiga, ukuran

kernel, ukuran *stride*, dan ukuran padding adalah 3x3, 1, dan 1. Rincian dari backbone ResNet18 dapat dilihat pada Tabel 3 (B. Li & Gu, 2023).

Tabel 3. Detail ResNet-18 Backbone

Stage	Output	Structure Details
Conv1	$112 \times 112 \times 64$ $56 \times 56 \times 64$	$7 \times 7, 64, s = 2, p = 3$ $3 \times 3, \text{max-pooling}, s = 2, p = 1$
Layer1	$56 \times 56 \times 64$	$\begin{bmatrix} 3 \times 3.64 \\ 3 \times 3.64 \end{bmatrix} \times 2$
Layer2	$28 \times 28 \times 128$	$\begin{bmatrix} 3 \times 3.128 \\ 3 \times 3.128 \end{bmatrix} \times 2$
Layer3	$14 \times 14 \times 256$	$\begin{bmatrix} 3 \times 3.256 \\ 3 \times 3.256 \end{bmatrix} \times 2$
Layer4	$7 \times 7 \times 512$	$\begin{bmatrix} 3 \times 3.512 \\ 3 \times 3.512 \end{bmatrix} \times 2$
Pool	$1 \times 1 \times 512$	Global average pooling

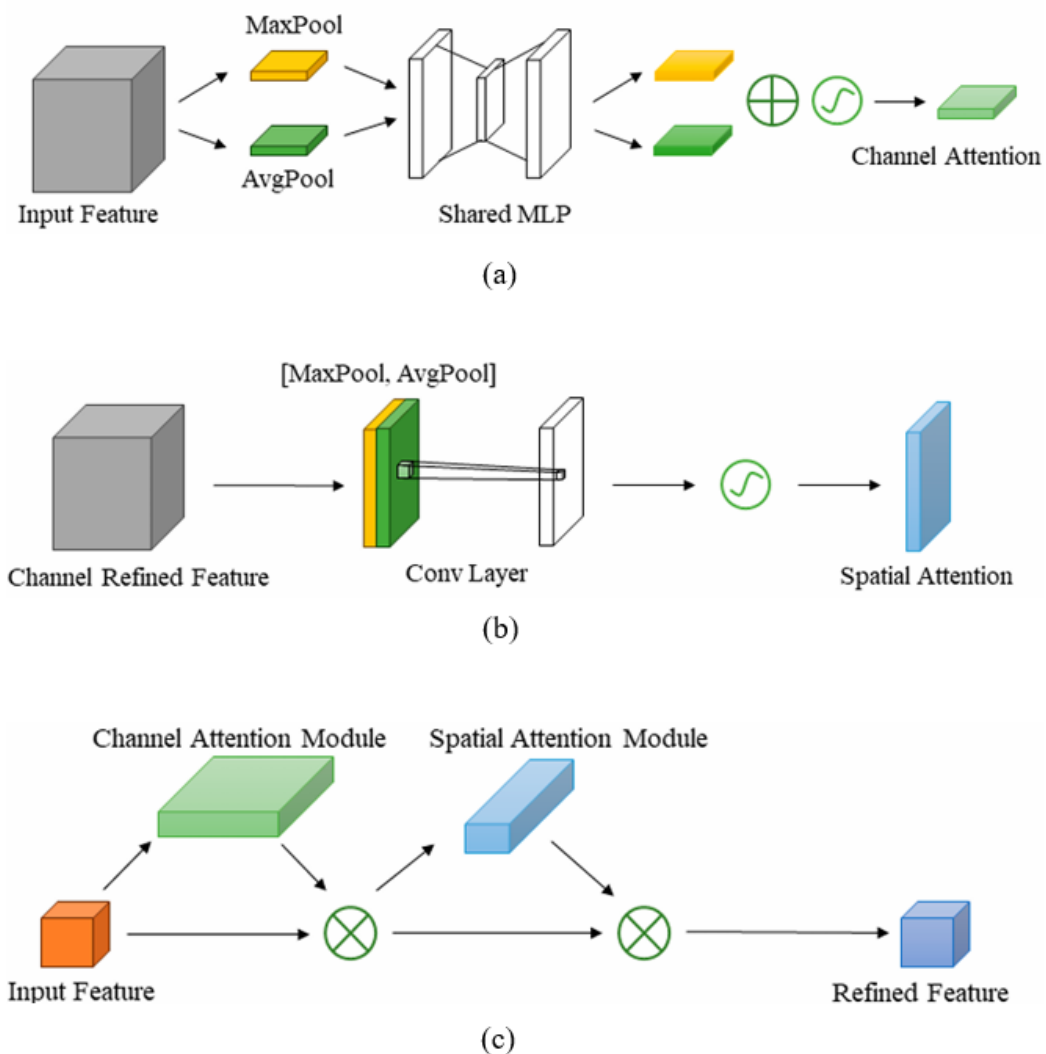
Sumber : (B. Li & Gu, 2023)

2.9 Convolutional Block Attention Module (CBAM)

Convolutional Block Attention Module (CBAM) bertujuan untuk memperoleh informasi kontekstual yang ada dalam gambar guna menangkap relevansi dan membantu model memprioritaskan area penting sambil mengabaikan informasi yang tidak relevan. Mekanisme ini mencakup *channel attention module* (*channel attention module*) dan *spatial attention module* (*spatial attention module*), yang masing-masing menekankan pentingnya saluran fitur dan area spasial dalam gambar. *Channel attention module*, yang dikenal sebagai *channel attention module* (CAM), membantu mengidentifikasi saluran fitur yang penting untuk tugas tertentu dengan menganalisis hubungan antara saluran yang berbeda dan mengoptimalkan alokasi peta fitur, sehingga meningkatkan kinerja model (H. Li et al., 2023).

Di sisi lain, *spatial attention module*, yang dikenal sebagai SAM, fokus pada menentukan pentingnya area *pixel* dalam gambar. Modul ini memfasilitasi pemahaman yang lebih baik tentang area yang ada, terutama untuk mengekstraksi fitur tepi secara akurat. CBAM memperkenalkan kedua *attention mechanisms* ini dan menggunakan struktur berurutan dari *channel attention* dan *spatial attention*, dengan mempertimbangkan analisis dimensi *channel* dan ruang. Hal ini memungkinkan jaringan saraf untuk memproses fitur gambar dengan cermat, sambil memperhatikan informasi pada berbagai skala. Selain itu, CBAM dikenal karena sifatnya yang ringan dan integrasi yang mulus ke dalam berbagai jaringan

saraf, sehingga meningkatkan fleksibilitas dan kinerja model. Kesimpulannya, penggunaan *channel attention* memungkinkan model untuk fokus pada informasi penting, yang mengarah pada ekstraksi fitur tepi yang lebih presisi dan efisien, sehingga meningkatkan kinerja model dan pemahaman terhadap gambar (H. Li et al., 2023).



Gambar 21 Struktur CBAM: (a) *Channel Attention Module*; (b) *Spatial Attention Module*; (c) CBAM

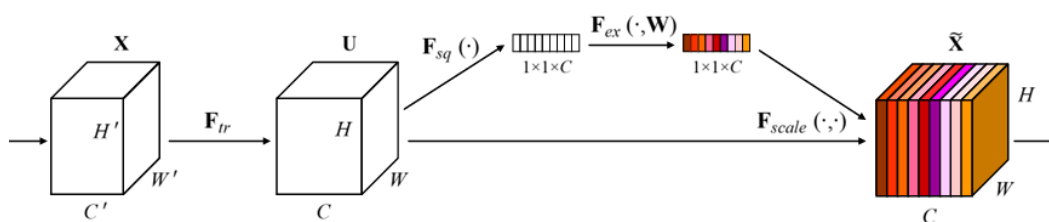
Sumber: (H. Li et al., 2023)

Struktur CBAM, seperti yang diilustrasikan pada gambar 21, terdiri dari *channel attention module* dan *spatial attention module*. Dalam *channel attention module*, *pooling* diterapkan pada peta fitur input untuk memperoleh informasi bobot untuk setiap saluran. Kemudian, informasi bobot ini diteruskan ke *spatial attention module*. Selanjutnya, *spatial attention module* menggunakan operasi *pooling* maksimum dan rata-rata pada nilai fitur dari setiap titik spesifik di seluruh saluran

pada peta fitur input. Operasi ini secara efektif menangkap fitur pada berbagai skala. Bobot untuk setiap titik fitur pada peta fitur input kemudian diperoleh menggunakan prosedur yang sama dengan yang digunakan dalam *channel attention module*. Terakhir, bobot yang diperoleh digunakan untuk melakukan konvolusi berbobot pada peta fitur input asli, menghasilkan fitur dalam yang mengintegrasikan informasi kontekstual multi-skala. Singkatnya, struktur CBAM, melalui *channel attention module* dan perhatian spasialnya, memungkinkan ekstraksi fitur multi-skala pada peta fitur input. Peningkatan ini menghasilkan pemahaman yang lebih baik tentang konten gambar dan meningkatkan kemampuan analisis serta pengenalan model (H. Li et al., 2023).

2.10 Squeeze-and-Excitation (SE) Block

Struktur blok *Squeeze-and-Excitation* (SE) digambarkan pada Gambar 22. Untuk setiap transformasi F_{tr} yang memetakan input X ke peta fitur U dengan $U \in \mathbb{R}^{H \times W \times C}$, seperti konvolusi, kita dapat membangun blok SE yang sesuai untuk melakukan kalibrasi ulang fitur. Fitur U pertama kali melewati operasi *squeeze*, yang menghasilkan deskriptor saluran dengan mengagregasi peta fitur di seluruh dimensi spasialnya ($H \times W$). Fungsi deskriptor ini adalah menghasilkan *embedding* dari distribusi global respons fitur berdasarkan saluran, sehingga informasi dari lapangan reseptif global jaringan dapat digunakan oleh semua lapisannya. Agregasi ini diikuti oleh operasi eksitasi, yang berupa *self-gating mechanisms* sederhana yang menggunakan *embedding* sebagai input dan menghasilkan kumpulan bobot modulasi per saluran. Bobot ini diterapkan pada embedding peta fitur U untuk menghasilkan output dari blok SE yang dapat langsung diumpungkan kelapisan jaringan berikutnya (Hu et al., 2017).



Gambar 22 *Squeeze-and-Excitation block*

Sumber: (Hu et al., 2017).

Kita dapat membangun *SE network* (SENet) dengan hanya menumpuk sejumlah blok SE. Selain itu, blok SE ini juga dapat digunakan sebagai pengganti langsung untuk blok asli pada berbagai kedalaman dalam *network architecture*. Meskipun template blok bangunan ini bersifat generik, peran yang dilakukannya berbeda pada berbagai kedalaman dalam jaringan. Pada lapisan awal, blok SE mengeksitasi fitur informatif dengan cara yang tidak bergantung pada kelas, memperkuat representasi tingkat rendah yang dibagikan. Pada lapisan selanjutnya, blok SE menjadi semakin spesifik untuk kelas, dan merespons input yang berbeda dengan cara yang sangat spesifik untuk kelas tersebut. Akibatnya, manfaat dari kalibrasi ulang fitur yang dilakukan oleh blok SE dapat terakumulasi di seluruh *channel* (Hu et al., 2017).

2.10.1 Squeeze: Global Information Embedding

Untuk menangani masalah eksploitasi *channel dependency*, pertama-tama mempertimbangkan sinyal ke setiap *channel* dalam output fitur. Setiap filter yang dipelajari beroperasi dengan bidang reseptif lokal dan akibatnya, setiap unit dari output transformasi U tidak dapat memanfaatkan informasi kontekstual di luar wilayah ini. Untuk mengatasi masalah ini, maka memadatkan *global spatial information* menjadi deskriptor *channel*. Ini dicapai dengan menggunakan *pooling* rata-rata global untuk menghasilkan statistik *channel* (Hu et al., 2017). Secara formal, statistik dihasilkan dengan mengecilkan U melalui dimensi spasialnya $H \times W$, sehingga elemen ke- c dari z dihitung sebagai:

$$z_c = \mathbf{F}_{sq}(\mathbf{u}_c) = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W u_c(i, j). \quad (4)$$

Output dari transformasi U dapat diinterpretasikan sebagai kumpulan deskriptor lokal yang statistiknya ekspresif untuk seluruh gambar. Memanfaatkan informasi semacam itu sudah umum dalam pekerjaan rekayasa fitur sebelumnya (Hu et al., 2017).

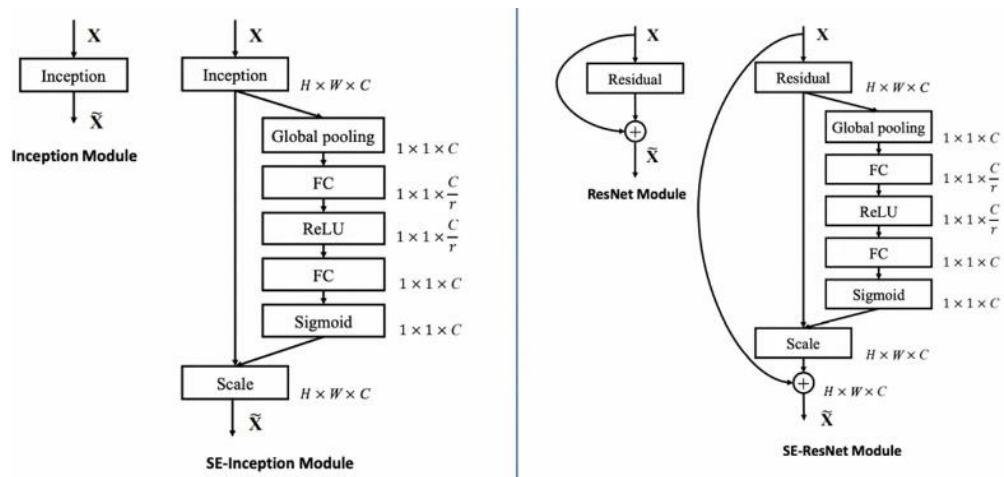
2.10.2 Excitation: Adaptive Recalibration

Untuk memanfaatkan informasi dari operasi *squeeze*, selanjutnya dengan operasi kedua yang bertujuan menangkap ketergantungan antar saluran. Fungsi ini harus fleksibel untuk mempelajari interaksi non-linear antar saluran dan memungkinkan beberapa saluran ditekankan sekaligus. Dengan menggunakan mekanisme gating sederhana dengan aktivasi *sigmoid* yang diwakili oleh:

$$\mathbf{s} = \mathbf{F}_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(g(\mathbf{z}, \mathbf{W})) = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \mathbf{z})), \quad (5)$$

Dimana σ adalah fungsi ReLU, $W_1 \in \mathbb{R}^{\frac{C}{r} \times C}$ dan $W_2 \in \mathbb{R}^{C \times \frac{C}{r}}$. Mekanisme ini mencakup dua lapisan *fully-connected* dengan rasio reduksi R , ReLU, dan lapisan peningkatan dimensi kembali ke dimensi saluran dari output transformasi U (Hu et al., 2017).

Operator eksitasi memetakan deskriptor input z ke bobot saluran, memperkenalkan dinamika yang dikondisikan oleh input dan berfungsi sebagai *self-attention* pada saluran yang tidak terbatas pada lapangan reseptif lokal. Strategi ini meningkatkan sensitivitas jaringan terhadap fitur informatif yang dapat dieksploitasi oleh transformasi berikutnya, sehingga mengoptimalkan kinerja model secara keseluruhan (Hu et al., 2017).



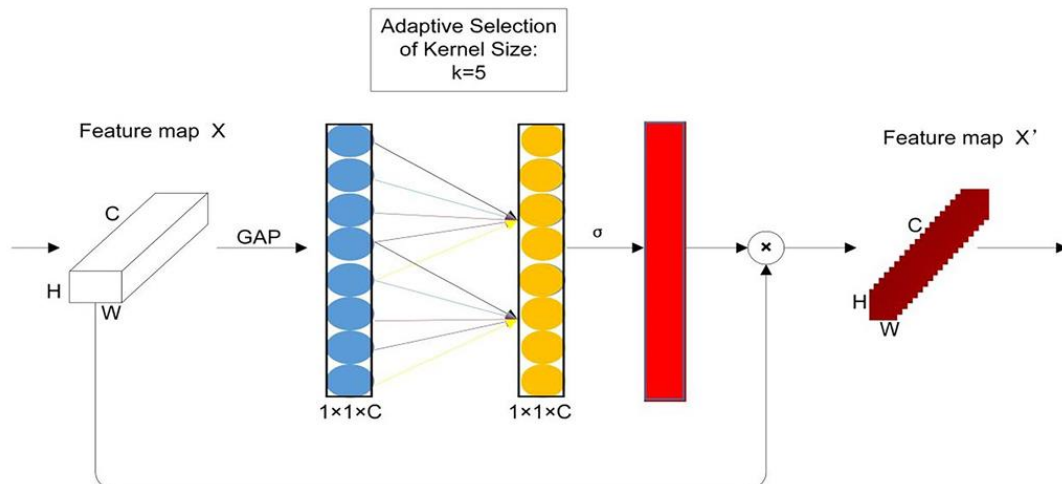
Gambar 23 Penerapan SE block pada modul Inception (kiri) dan ResNet (kanan)
Sumber: (Hu et al., 2017).

Gambar 23 menunjukkan bagaimana blok SE yang sama diterapkan tepat setelah volume keluaran lapisan konvolusional, untuk memperkaya representasinya sebelum memindahkannya ke lapisan berikutnya. Dalam hal ini, modul Inception

dan ResNet ditampilkan, memperlihatkan bagaimana blok SE dapat diintegrasikan ke dalam berbagai *neural network architecture* yang berbeda. Pada modul *Inception*, blok SE disisipkan setelah setiap modul *Inception*, sehingga memungkinkan penggabungan informasi global dan penekanan fitur penting di seluruh saluran. Sedangkan pada ResNet, blok SE diterapkan pada cabang non-identitas dari modul residual, yang memberikan peningkatan dalam memanfaatkan ketergantungan antar saluran sebelum penjumlahan dengan cabang (Hu et al., 2017).

2.11 *Efficient channel attention (ECA) Layer*

Modul *efficient channel attention (ECA)* (Wang et al., 2020) adalah strategi interaksi *cross channel* lokal tanpa reduksi dimensi, yang dapat diimplementasikan secara efisien melalui konvolusi satu dimensi (1D). Modul ECA dikembangkan dengan meningkatkan *Squeeze-and-Excitation (SE)* (Hu et al., 2020), yang merupakan metode *channel attention learning* yang efektif. Modul ini memprediksi bobot yang akan diterapkan pada setiap *channel* output. Metode SE pertama-tama menggunakan *global average pooling (GAP)* untuk setiap *channel* fitur secara individual untuk mereduksi *channel* fitur dua dimensi menjadi angka *real*. Kemudian, dua lapisan *fully-connected* menangkap interaksi *cross channel non-linear*. Akhirnya, fungsi *Sigmoid* menghasilkan bobot *channel* dengan nilai antara 0 dan 1. Bobot ini ditambahkan ke *feature channel* sebagai bobot untuk menghasilkan level input data berikutnya. Karakteristik SE adalah menggunakan korelasi antar *channel* alih-alih korelasi dalam distribusi spasial. Dengan mengendalikan besarnya bobot, fitur penting diperkuat dan fitur tidak penting dilemahkan sehingga fitur yang diekstraksi lebih terarah. Dibandingkan dengan SE, peningkatan ECA adalah bahwa operasi GAP pada *feature channel* tidak mengurangi dimensi. Sebaliknya, modul ini menangkap informasi *cross-channel interaction* dengan mempertimbangkan setiap saluran dan K tetangga terdekatnya. Modul ECA dapat digunakan sebagai modul *plug-and-play* yang sangat ringan untuk meningkatkan kinerja berbagai CNN (Gao et al., 2020; Wang et al., 2020). Proses implementasinya ditunjukkan pada Gambar 24.



Gambar 24 *Efficient channel attention module*

Sumber: (Yuan et al., 2022)

Cross-channel interaction adalah metode kombinasi fitur baru yang dapat meningkatkan ekspresi fitur dari semantik tertentu. Setelah *global average pooling* pada peta fitur, modul ECA menangkap *Cross-channel interaction* lokal dengan mempertimbangkan interaksi antara setiap fitur *channel* dan k *neighbor channel*. *Cross-channel interaction* dapat menghindari reduksi dimensi melalui operasi konvolusi satu dimensi dan mewujudkan interaksi *multi-channel* secara efektif (Gao et al., 2020). Oleh karena itu, bobot dari fitur y dapat dihitung sebagai:

$$w_i = \sigma \left(\sum_{j=1}^k \alpha^j y_i^j \right), y_i^j \in \Omega_i^k \quad (6)$$

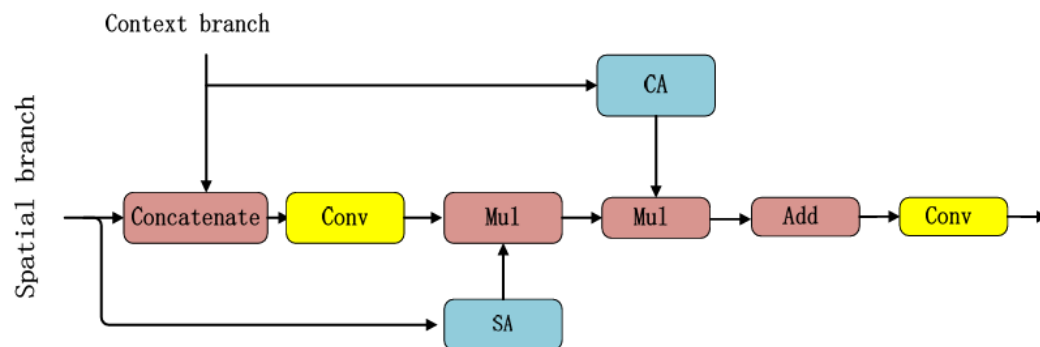
Di mana, σ merepresentasikan fungsi *Sigmoid*, Ω_i^k adalah kumpulan k *neighbor channel* dari y , dan rumus (6) menunjukkan bahwa semua *channel* berbagi parameter kemiringan yang sama, sehingga memungkinkan efisiensi model yang lebih tinggi (Gao et al., 2020).

$$w = \sigma(\text{C1D}_k(y)) \quad (7)$$

Rumus (7) merepresentasikan realisasi fitur *channel* melalui *cross-channel one-dimensional convolution* (C1D), di mana C1D merepresentasikan konvolusi satu dimensi dan k merepresentasikan cakupan *local cross channel interaction*, yaitu berapa banyak tetangga di sekitar *channel* yang berpartisipasi dalam prediksi *attention channel* ini (Gao et al., 2020).

2.12 Feature Cross Attention (FCA)

feature cross-attention (FCA) module menggunakan *spatial attention module* untuk mengekstraksi informasi spasial kemudian menggunakan *channel attention mechanism* untuk menangkap informasi kontekstual. Model FCA ditunjukkan pada Gambar 25. *Output features* dari kedua cabang ini berbeda. *High-level features*, yang terutama terdiri dari informasi kategori, dapat diekstraksi menggunakan *channel attention module*. Sebaliknya, *low-level features* yang lebih banyak berhubungan dengan informasi spasial tidak dapat langsung diambil sampelnya dan digabungkan. Kita dapat menggunakan *spatial attention module* untuk mengekstraksi fitur dari *low-level features* (Zeng et al., 2020).



Gambar 25 Feature Cross Attention Module (FCA)

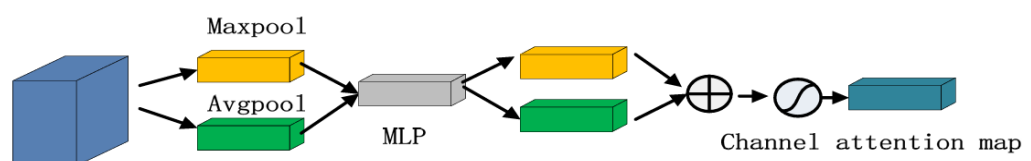
Sumber: (Zeng et al., 2020)

High-level features dari *channel attention module* pada FCA yang ditambahkan digunakan untuk menyediakan informasi kontekstual, sementara *low-level features* yang diekstraksi oleh *spatial attention module* digunakan untuk memperhalus pemetaan *pixel*. Pertama-tama menggabungkan *output features* dari kedua cabang, kemudian melakukan konvolusi, *batch normalization*, dan pemrosesan unit ReLU pada fitur gabungan tersebut. Selanjutnya, fitur yang digabungkan oleh SA module dan output dari *spatial branch* digunakan sebagai input untuk membantu memperhalus pemetaan. Fitur dari SA module dinormalisasi dan dikonvolusi secara non-linear berbentuk *S*, kemudian dikalikan dengan fitur yang digabungkan. *Context peak output space* diterapkan pada *channel attention block*, dan fitur kontekstual dikompresi sepanjang dimensi spasial melalui *global pooling* dan *maximum pooling* untuk memperoleh dua vektor. Kemudian, kedua vektor tersebut dibagikan ke *fully connected layer* dan *Sigmoid operator* untuk

menghasilkan *attention graph*, dan akhirnya dilakukan konvolusi, *batch normalization*, dan fusi unit ReLU (Zeng et al., 2020).

2.12.1 Channel attention module

Attention mechanism telah digunakan secara luas dalam berbagai bidang *deep learning* dalam beberapa tahun terakhir, dan telah menunjukkan performa yang baik dalam *image processing*, *speech recognition*, dan *natural language processing*. SEnet mempelajari bobot fitur sesuai dengan *loss* melalui jaringan, memberikan peta fitur penting bobot besar, dan peta fitur tidak valid atau tidak penting bobot kecil untuk melatih model agar mencapai hasil yang lebih baik. SEnet diintegrasikan ke dalam beberapa *original classification network* dengan menambahkan sedikit parameter dan perhitungan. Metode ekstraksi *attention of feature channels* pada dasarnya mirip dengan SEnet. Metode ekstraksi fitur menggunakan *maxpool* ditambahkan pada SEnet. Output akhir adalah hasil penjumlahan dari hasil *pooling* rata-rata dan hasil *pooling* maksimum. Metode ekstraksi fitur dengan *avgpool* sama dengan metode ekstraksi *avgpool* pada SEnet. Selain itu, saat menggunakan kedua *pool* ini, digunakan MLP bersama untuk menghemat parameter, dan kedua fitur saluran yang diagregasi berada dalam ruang *embedding* semantik yang sama (Zeng et al., 2020). Dengan menggunakan modul perhatian saluran seperti yang ditunjukkan pada Gambar 26.



Gambar 26 Channel attention module (CA)

Sumber: (Zeng et al., 2020)

Setiap *channel* dari *channel attention module* berfungsi sebagai detektor khusus. Ini penting untuk *channel attention module* agar dapat fokus pada jenis fitur tertentu. Untuk merangkum karakteristik spasial, digunakan dua metode, yaitu *global average pooling* dan *maximum pooling*, untuk memanfaatkan informasi yang berbeda (Zeng et al., 2020). Proses operasi *channel attention module* ditunjukkan dalam rumus 8 dan 9:

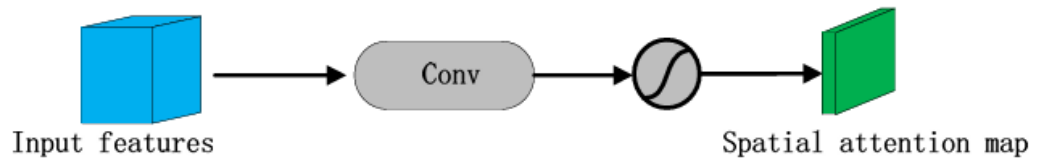
$$M_c = \sigma \left(MLP \left(AvgPool \left(F \right) \right) + MLP \left(MaxPool \left(F \right) \right) \right) \quad (8)$$

$$= \sigma \left(W_1 \left(W_0 \left(F_{avg}^c \right) \right) + W_1 \left(W_0 \left(F_{max}^c \right) \right) \right) \quad (9)$$

Dimana MLP adalah *multilayer perception*; σ adalah fungsi aktivasi *sigmoid*. Inputnya adalah fitur $F \in \mathbb{R}^{H \times W \times C}$. Pertama, dilakukan *global average pooling* dan *maximum pooling* dari ruang untuk mendapatkan dua deskripsi saluran $1 \times 1 \times C$. Kemudian, deskripsi ini dikirim ke *two-layer neural network*, di mana jumlah neuron di lapisan pertama adalah C/r , fungsi aktivasinya adalah ReLU, dan jumlah neuron di lapisan kedua adalah C . *two-layer neural network* ini digunakan bersama. Kemudian, kedua fitur tersebut dijumlahkan dan menghasilkan koefisien bobot M_c melalui fungsi aktivasi *sigmoid*. Akhirnya, koefisien bobot ini dikalikan dengan fitur asli F untuk mendapatkan fitur baru yang diskalakan (Zeng et al., 2020).

2.12.2 Spatial attention module

Spatial attention module berfokus pada lokasi fitur-fitur yang bermakna. Diagram modelnya ditunjukkan pada Gambar 27.



Gambar 27 *Spatial attention module* (SA)

Sumber: (Zeng et al., 2020)

Serupa dengan *channel attention*, diberikan fitur $F \in \mathbb{R}^{H \times W \times C}$, pertama-tama dilakukan *average pooling* dan *maximum pooling* pada satu dimensi *channel* untuk mendapatkan dua deskripsi saluran $H \times W \times 1$. kemudian menggabungkan kedua deskripsi tersebut berdasarkan *channel*. Setelah itu, melewati lapisan konvolusi dengan ukuran kernel konvolusi 7×7 , fungsi aktivasinya adalah *Sigmoid*, dan koefisien bobot M_s diperoleh. Akhirnya, mengalikan koefisien bobot dan fitur F' menghasilkan fitur baru (Zeng et al., 2020). Proses operasinya ditunjukkan dalam rumus 10 & 11:

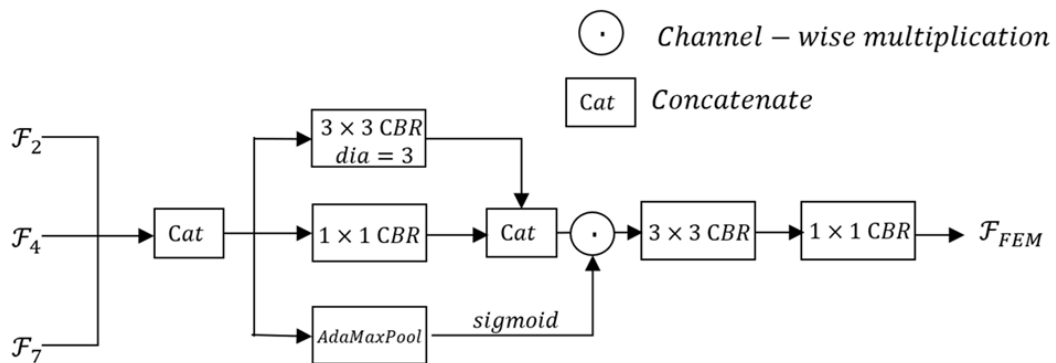
$$M_s(F) = \sigma \left(f^{7 \times 7} \left([AvgPool \left(F \right); MaxPool \left(F \right)] \right) \right) \quad (10)$$

$$= \sigma \left(f^{7 \times 7} \left([F_{avg}^s; F_{max}^s] \right) \right) \quad (11)$$

Di mana σ adalah fungsi aktivasi *sigmoid*; f adalah lapisan konvolusi; $[\cdot]$ adalah penggabungan peta fitur dalam channel dimension (Zeng et al., 2020).

2.13 Feature Enhancement Module (FEM)

Feature enhancement module (FEM) diusulkan untuk menggabungkan berbagai tingkat fitur guna menghasilkan peta fitur dengan detail dan semantik yang cukup (Yao et al., 2022). Gambar 28 menunjukkan arsitektur FEM.



Gambar 28 Arsitektur *Feature enhancement module* (FEM)

Sumber: (Yao et al., 2022)

Secara spesifik, input FEM adalah *high-level feature* $f_7 \in \mathbb{R}^{C_7 \times H_7 \times W_7}$, *middle-level feature* $f_4 \in \mathbb{R}^{C_4 \times H_4 \times W_4}$, dan *bottom-level feature* $f_2 \in \mathbb{R}^{C_2 \times H_2 \times W_2}$. Karena $f_2, f_4,$ dan f_7 memiliki resolusi yang berbeda, $f_4,$ dan f_7 diskalakan ke ukuran peta fitur yang sama dengan f_2 melalui interpolasi bilinear. Kemudian, $f_2, f_4,$ dan f_7 digabungkan untuk menghasilkan peta fitur $f_{concat} \in \mathbb{R}^{(C_2 + C_4 + C_7) \times H_2 \times W_2}$ (Yao et al., 2022). Proses ini dapat direpresentasikan dengan rumus berikut:

$$f_{concat} = cat(f_2, f_4, f_7) \quad (12)$$

di mana 'cat' merujuk pada operasi penggabungan sepanjang arah *channel*. Jumlah informasi yang termasuk dalam setiap lapisan tidak terpengaruh oleh operasi cat di sini. Oleh karena itu, fitur hasil f_{concat} akan mengandung informasi detail dan semantik yang berbeda. Kemudian, studi ini melakukan dua operasi berikut pada f_{concat} :

$$f_{branch} = \text{cat}(w_1 * f_{concat}, w_2 * f_{concat}) \quad (13)$$

$$f_{FEM} = \text{sigmoid}(\text{AdaMaxPool}(f_{concat}))f_{branch} \quad (14)$$

di mana '*' merujuk pada operasi konvolusi, w_1 adalah lapisan CBR dengan kernel 1x1, w_2 adalah lapisan CBR dengan kernel 3x3 yang mengalami dilasi dengan tingkat dilasi 3, dan *AdaMaxPool* merujuk pada operasi *adaptive maximum pooling operation*. Secara spesifik, w_1 digunakan untuk memperkuat koneksi setiap *channel* sehingga informasi *branch* dapat mengalir ke *channel* lain. Fungsi w_2 adalah untuk memperbesar bidang reseptif f_{concat} guna memperoleh lebih banyak informasi global. Fitur dari kedua *branch* yang disebutkan di atas digabungkan secara sederhana dengan operasi *cat* untuk menyelesaikan komplementasi informasi. Output yang dihasilkan ditandai sebagai $f_{FEM} \in \mathbb{R}^{(C_2 + C_4 + C_7) \times H_2 \times W_2}$. Untuk merekalibrasi f_{branch} agar fokus pada *informative channel*, *AdaMaxPool* dilakukan pada untuk memperoleh $f_{AdaMaxPool} \in \mathbb{R}^{(C_2 + C_4 + C_7) \times 1 \times 1}$. Fungsi aktivasi *sigmoid* kemudian diterapkan untuk menskalakan ke rentang [0, 1]. Selanjutnya, $f_{AdaMaxPool}$ dan f_{branch} dikalikan sepanjang dimensi *channel*-nya. Fitur yang dihasilkan didefinisikan sebagai $f_{FEM} \in \mathbb{R}^{(C_2 + C_4 + C_7) \times H_2 \times W_2}$ (Yao et al., 2022).

Akhirnya, lapisan CBR 3x3 dan lapisan CBR 1x1 dilakukan pada f_{FEM} lapisan CBR 3x3 bertujuan untuk mengurangi jumlah *channel* dari $C_2 + C_4 + C_7$ menjadi C (di mana $C=256$), dan memperbesar bidang reseptif. Lapisan CBR 1x1 berusaha untuk meningkatkan konektivitas *channel*, yang dapat lebih mendorong aliran informasi antara *channel*. FEM membawa total $5(C_2 + C_4 + C_7)^2 + 9C(C_2 + C_4 + C_7) + C^2$ parameter baru, yang setara dengan 0,76 juta—artinya, FEM hanya menambah sejumlah kecil parameter pada backbone. Namun, FEM meningkatkan beban komputasi pada backbone hingga batas tertentu, karena sebagian besar komputasi terjadi pada peta fitur resolusi tinggi, yang mungkin menurunkan kecepatan inferensi (Yao et al., 2022).

2.14 Loss Function

Pada dataset yang ada, biasanya *background* mendominasi gambar gambar yang ada, sehingga data menjadi tidak seimbang. Untuk menghindari *network*

berfokus pada *background*, sangat penting untuk memilih fungsi *loss* yang tepat (Yao et al., 2022). Dengan menggabungkan *Focal loss*, *Dice loss* dan *cross entropy loss* sebagai fungsi *loss* akhir maka:

$$loss = loss_{FL} + loss_{De} + loss_{Ce} \quad (15)$$

Focal loss memiliki performa yang sangat baik dalam menangani ketidakseimbangan kelas antara *foreground* dan *background* (Yao et al., 2022), yang didefinisikan sebagai berikut:

$$loss_{FL} = -\alpha\gamma(1-p)^\gamma \cdot \log(p) - (1-\alpha)(1-y)p^\gamma \cdot \log(1-p) \quad (16)$$

di mana γ adalah konstanta yang merepresentasikan parameter fokus. *Dice loss* menghitung rasio perpotongan antara nilai prediksi dan *ground truth* (Yao et al., 2022), yang didefinisikan sebagai berikut:

$$loss_{De} = 1 - \frac{2|X \cap Y|}{|X| + |Y|} \quad (17)$$

Di mana X merepresentasikan *ground truth* dan Y merepresentasikan prediksi (Yao et al., 2022). *Cross-entropy loss* terutama mengukur perbedaan akurasi antara *pixel* (Y. Gao et al., 2023). *Cross-entropy loss* dinyatakan sebagai berikut:

$$loss_{Ce} = -\frac{1}{N} \sum_{I=0}^{N-1} \sum_{K=0}^{K-1} y_{i,k} \ln p_{i,k}, \quad (18)$$

di mana N adalah jumlah sampel, K merepresentasikan jumlah kategori, $p_{i,k}$ merepresentasikan probabilitas prediksi dari kategori K_{th} dalam sampel I_{th} , dan $y_{i,k}$ menunjukkan apakah hasil prediksi sesuai dengan label sebenarnya (Y. Gao et al., 2023).

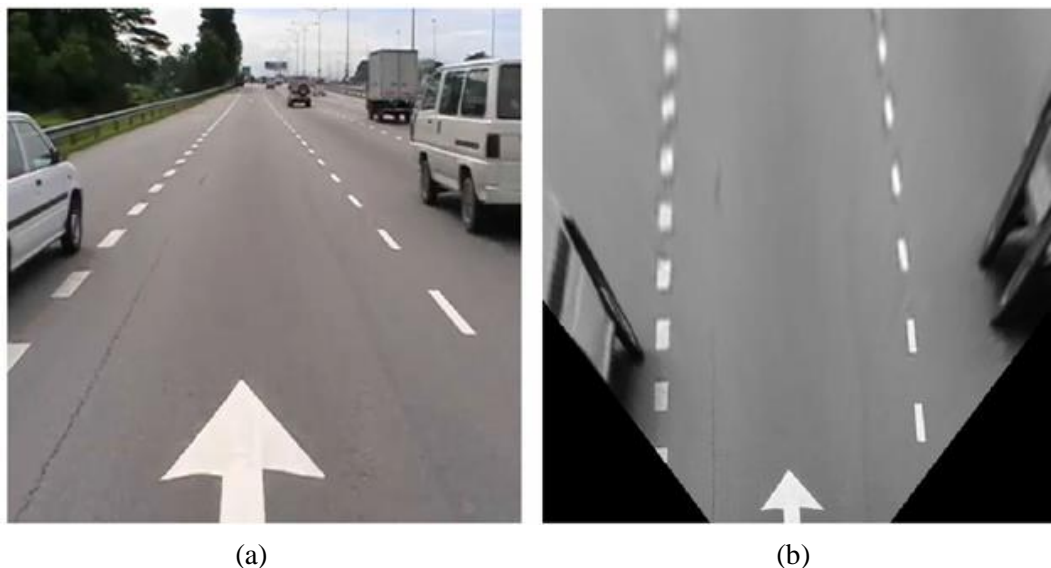
2.15 Inverse Perspective Mapping (IPM)

Inverse Perspective Mapping (IPM) adalah sebuah teknik transformasi yang digunakan untuk memetakan objek 3D yang diproyeksikan ke dalam bidang 2D ke posisi baru untuk menciptakan gambar 2D baru. IPM bekerja dengan mengubah perspektif citra untuk menghilangkan distorsi perspektif dan memperbaiki ketajaman gambar. Konsep dasar IPM melibatkan pemetaan titik pada bidang citra ke titik pada objek 3D dengan mempertimbangkan titik lenyap (*vanishing point*) pada citra perspektif. IPM digunakan untuk mengoreksi distorsi citra yang disebabkan oleh sudut bidikan kamera dan memperbaiki ketajaman gambar. Teknik

ini juga mempertimbangkan kemiringan bidang citra untuk mengoreksi distorsi yang dihasilkan oleh kemiringan bidang tersebut (Yoo & Lee, 2023).

IPM telah diterapkan dalam berbagai bidang, termasuk navigasi *autonomous car*. Dalam konteks *autonomous car*, IPM telah berkembang menjadi teknik yang digunakan untuk mengidentifikasi garis jalan, jalur lalu lintas, dan struktur jalan lainnya dalam citra kamera, serta untuk membuat keputusan navigasi yang tepat. Penggunaan IPM dalam sistem navigasi *autonomous car* membantu dalam meningkatkan persepsi lingkungan sekitar, mengidentifikasi rintangan, dan merencanakan rute dengan lebih baik berdasarkan gambaran yang lebih akurat dari lingkungan sekitar kendaraan (Yoo & Lee, 2023).

Tantangan utama dalam menerapkan IPM dalam konteks *autonomous car* adalah ketika jalur tidak selalu *linear* atau *paralel*, yang dapat menyebabkan kesulitan dalam deteksi lajur yang akurat. Selain itu, perubahan parameter kamera eksternal seperti sudut *pitch* akibat getaran kamera juga dapat mempengaruhi kinerja IPM. Tren terbaru dalam pengembangan teknik-teknik IPM termasuk penggunaan algoritma adaptif untuk menyesuaikan parameter kamera dan deteksi titik hilang, serta penerapan teknologi seperti *Convolutional Neural Networks* (CNN) untuk meningkatkan citra jalan (Ozgunalp, 2019).



Gambar 29 IPM: a) *input* gambar jalanan; b) *output* gambar jalan setelah penerapan IPM

Sumber: (Ozgunalp, 2019)

Ketika gambar diambil melalui kamera pada gambar 29, informasi dari dunia 3D diproyeksikan ke dalam gambar 2D, yang menyebabkan kehilangan sebagian informasi. Untuk mengekstrak koordinat 3D $[x, y, z]$ dari suatu *pixel*, IPM beroperasi dengan asumsi bahwa jalan adalah datar dan parameter kamera (baik eksternal maupun internal) diketahui. Dalam konteks ini, semua *pixel* pada gambar dianggap sebagai bagian dari jalan (Ozgunalp, 2019). Kemudian, menggunakan geometri, persamaan yang sesuai dapat digunakan untuk pemrosesan lebih lanjut seperti pada rumus berikut.

$$z(u, v) = h x \cot\left([\theta - \alpha] + u \frac{2\alpha}{n-1}\right) x \cos\left([\gamma - \alpha] + v \frac{2\alpha}{n-1}\right) + L \quad (19)$$

$$x(u, v) = h x \cot\left([\theta - \alpha] + u \frac{2\alpha}{n-1}\right) x \sin\left([\gamma - \alpha] + v \frac{2\alpha}{n-1}\right) + d \quad (20)$$

$$y = 0 \quad (21)$$

Keterangan:

h = tinggi kamera dari bidang

α = sudut yaw atau rotasi kamera sekitar sumbu vertikal

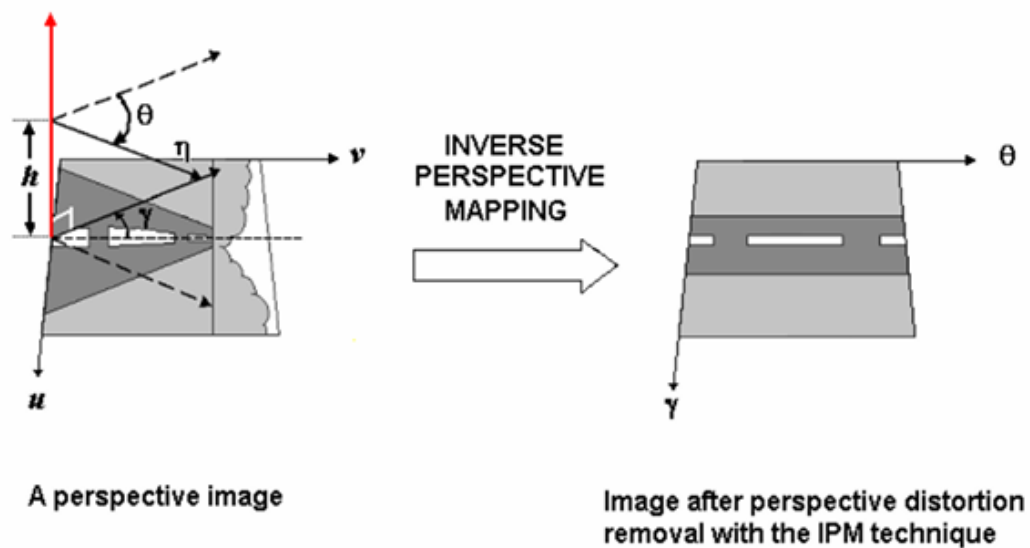
θ = sudut aperture atau sudut pandang kamera

d = offset lateral dari pusat kamera

L = offset pada sumbu z

n = resolusi gambar

u, v = koordinat *pixel* dalam citra



Gambar 30 Implementasi IPM

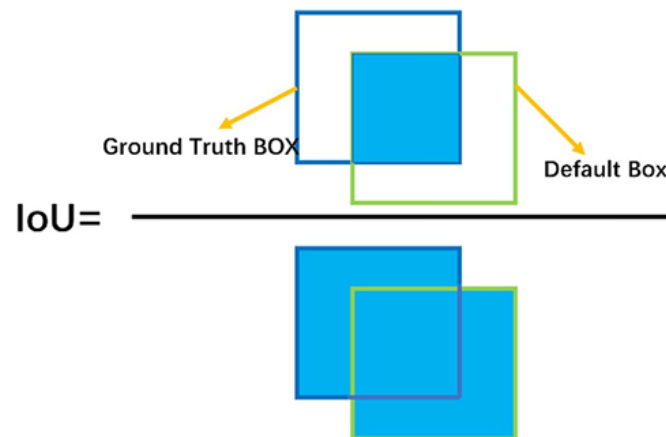
Sumber: (Muad et al., n.d., 2004)

Implementasi IPM dapat dilihat dalam gambar 30, yang menampilkan presentasi grafis dari proses implementasi IPM. Pada citra perspektif 2D, sudut yaw (J) mengalami perubahan dari negatif ke positif saat pemindaian dilakukan dari kiri ke kanan, sementara sudut kemiringan (T) berubah secara vertikal. Baik J maupun T berfungsi sebagai faktor 'bobot proyeksi' untuk semua *pixel* saat IPM melakukan proyeksi dan pemetaan ulang gambar perspektif (Muad et al., n.d., 2004)

2.16 Intersection over Union (IoU)

Intersection over union (IoU) yang juga dikenal sebagai *Indeks Jaccard* merupakan metrik evaluasi standar dalam deteksi objek dan segmentasi pada pengolahan citra. IoU mengukur seberapa besar area prediksi model dan *ground truth* tumpang tindih dibandingkan dengan total luas gabungan keduanya. Rumus matematik IoU adalah perbandingan dari luas area tumpang tindih terhadap luas area gabungan, dengan nilai antara 0 dan 1 yang menandakan tingkat tumpang tindih. IoU digunakan secara luas dalam berbagai konteks pengolahan citra, termasuk evaluasi deteksi objek dan *semantic segmentation*. Interpretasi IoU yang tinggi menunjukkan prediksi yang akurat, sementara IoU rendah menunjukkan ketidakakuratan. IoU digunakan untuk mengukur kualitas prediksi model dan memperbaiki kinerja model dengan mengurangi kesalahan. IoU menggambarkan

kemiripan antara dua bentuk sembarang dengan mempertimbangkan karakteristik bentuk objek seperti lebar, tinggi, dan lokasi kotak pembatas. Sifat inilah yang membuat IoU tetap konsisten terhadap perubahan yang ada (Rezatofighi et al., 2019).



Gambar 31 Ilustrasi skema IoU
Sumber: (Han et al., 2021)

Ilustrasi gambar 31 dapat dituliskan dalam persamaan berikut,

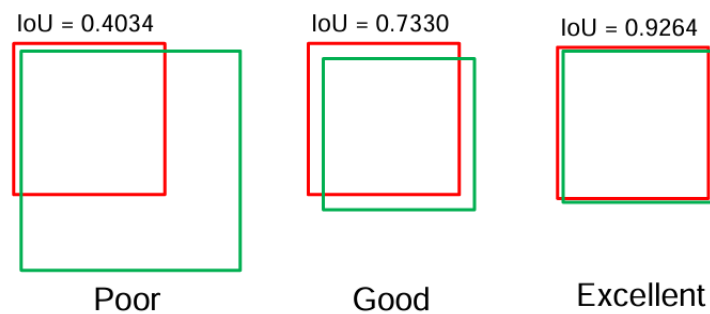
$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (22)$$

Keterangan:

A = Nilai Prediksi

B = Nilai Aktual

Berdasarkan persamaan di atas, matriks IoU memberikan penilaian kesesuaian antara dua buah himpunan. Cara menghitungnya yaitu dengan membandingkan luas area yang bertumpang tindih antara objek yang diprediksi oleh model dan objek sebenarnya, kemudian dibagi dengan total luas gabungan dari kedua objek tersebut. Nilai IoU yang tinggi menandakan bahwa model memiliki akurasi yang baik dalam memprediksi objek karena banyaknya kesesuaian dengan objek sebenarnya. Semakin mendekati 1 maka hasil yang didapatkan akan semakin baik, begitupun sebaliknya semakin mendekati 0 maka metrik IoU menandakan ketidakakuratan dalam memprediksi suatu objek. Sebagai contoh hasil penilaian IoU ditunjukkan pada gambar 32 berikut.



Gambar 32 Contoh hasil perhitungan IoU
 Sumber: (Pariwat & Seresangtakul, 2021)

Berdasarkan gambar 32 area objek aktual ditandai dengan warna hijau dan area objek prediksi ditandai dengan warna merah. Evaluasi IoU dilakukan dengan mengikuti standar tertentu. IoU pada bagian kiri sebesar 0,4034, menunjukkan kualitas yang rendah, sedangkan IoU di bagian tengah sebesar 0,7330, menandakan kualitas yang baik. Sementara itu, IoU yang tinggi sebesar 0,9264, menunjukkan kualitas yang sangat baik (Pariwat & Seresangtakul, 2021). Dalam evaluasi ini, nilai IoU digunakan untuk mengukur sejauh mana tumpang tindih antara area prediksi dan area sebenarnya, di mana nilai yang lebih tinggi mengindikasikan tingkat kesesuaian yang lebih baik antara prediksi model dan *ground truth*. Dengan demikian, IoU menjadi indikator yang penting dalam mengevaluasi kualitas prediksi dalam berbagai tugas pengolahan citra seperti deteksi objek dan *semantic segmentation*.

2.17 Root-Mean-Square Error (RMSE)

Metode paling umum untuk menggambarkan kemampuan prediktif suatu model adalah dengan menggunakan *root mean squared error* (RMSE). Metrik ini merupakan fungsi dari residu model, yang merupakan nilai-nilai yang diamati dikurangi dengan prediksi model. Nilai biasanya diinterpretasikan sebagai jarak rata-rata antara residu dan nol atau sebagai jarak rata-rata antara nilai-nilai yang diamati dan prediksi model (Kuhn & Johnson, 2013). Nilai 0 menunjukkan nilai terbaik, sedangkan nilai terburuk adalah $+\infty$. Persamaan untuk menghitung RMSE ditunjukkan pada persamaan 8 (Chicco et al., 2021).

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^n (X_i - Y_i)^2} \quad (23)$$

dimana,

N = jumlah total data

n = indeks iterasi

X_i = nilai aktual

Y_i = nilai prediksi