BAB 2
DASAR TEORI

2.1 KAJIAN PUSTAKA

Pada penelitian yang dilakukan oleh Hoanh Nguyen pada tahun 2019 dengan judul "*A Multi-Scale Deep learning Network For Vehicle Detection*", yang meneliti tentang metode pendeteksian kendaraan berbasis *deep learning* [11]. Penelitian ini menggunakan arsitektur VGG-16, dimana jaringan VGG-16 memiliki 16 *layer* dan termasuk model arsitektur yang sederhana. Untuk membandingkan keefektifan metode yang diusulkan dengan metode lain pada pendeteksian kendaraan, penelitian ini melakukan percobaan pada dataset KITTI. Hasil penelitian yaitu menunjukkan hasil deteksi dari metode yang diusulkan dan berbasis CNN canggih lainnya pada uji dataset KITTI. Metode yang diusulkan yaitu SSD, *Faster* R-CNN, MS-CNN, dan *Proposed method*. Metode yang diusulkan yaitu *deep learning multi*-skala, dimana peneliti merancang modul yang ditingkatkan yang memadukan lapisan konvolusi yang berbeda dengan menggunakan operasi dekonvolusi dan normalisasi. Pada tahap klasifikasi, peta fitur fusi dibuat untuk lapisan yang terhubung sepenuhnya dan wilayah kontekstual *multi*-skala dirancang untuk mengeksploitasi informasi di sekitarnya untuk proposal objek yang diberikan. Dari perbandingan tersebut hasil terbaik yaitu mendeteksi menggunakan MS-CNN. Namun, MS-CNN lebih lambat dibandingkan dengan metode yang diusulkan oleh penulis, dimana kinerja metode yang diusulkan hampir sama dengan MS-CNN sekaligus lebih cepat dan sederhana.

Pada penelitian yang dilakukan Xuesong Li, dkk pada tahun 2019 dengan judul "*Detection of Imaged Objects with Estimated Scales*", yang meneliti tentang deteksi objek berbasis CNN, yaitu menggunakan kinerja pendeteksian *Faster* R-CNN [12]. Penelitian ini menggunakan dataset KITTI, dengan metode *Faster* R-CNN dengan arsitektur jaringan VGG-16. Kami membandingkan dengan *Faster* R-CNN untuk keduanya Dalam uji KITTI, hasil *Faster* R-CNN dengan 70 *anchor* tersedia untuk semua objek, sedangkan *Faster* R-CNN dengan 9 *anchor* hanya dievaluasi ketika kendaraan terdeteksi. Untuk pendeteksian kendaraan, hasil *Faster* R-CNN dengan 70 dan 9 *anchor* dibandingkan dengan metode kami. Kami dapat menemukan bahwa meningkatkan jumlah *anchor* dapat sangat meningkatkan

akurasi deteksi. Performa pendeteksian *Faster* R-CNN dalam mendeteksi objek kendaraan mencapai puncaknya ketika jumlah *anchor* mencapai 70. Di sisi lain, metode perkiraan skala yang kami usulkan dapat mencapai akurasi deteksi serupa dalam kategori ringan. Namun, dalam kategori sedang dan sulit, skala peringkat dapat membantu menemukan ukuran *anchor* yang paling dekat dengan objek yang terdeteksi, sehingga mengurangi kesulitan menyimpang dari lokasi objek sedang dan sulit. *Faster* R-CNN dengan 70 *anchor* dievaluasi pada kumpulan data uji dan dibandingkan dengan metode kami. Peningkatan akurasi peta adalah 6,66% dan 0,68% dicapai untuk pengendara sepeda dan pejalan kaki, masing-masing. Salah satu penjelasannya adalah bahwa pengendara sepeda memiliki perbedaan skala dan rasio aspek yang jauh lebih besar daripada pejalan kaki, sehingga *anchor* yang disarankan memberikan lebih banyak manfaat daripada *anchor* yang telah ditentukan sebelumnya untuk pengendara sepeda.

Pada penelitian yang dilakukan oleh Hoanh Nguyen pada tahun 2019 dengan judul "*Improving Faster R-CNN Framework for Fast Vehicle Detection*" yang meneliti tentang meningkatkan kerangka kerja yang ditingkatkan berdasarkan *Faster* R-CNN untuk deteksi kendaraan cepat [4]. *Faster* R-CNN pada penelitian ini menggunakan arsitektur MobileNet. Hasilnya menunjukkan bahwa jaringan yang diusulkan sederhana, cepat, dan efisien. Selain itu, dibandingkan dengan metode deteksi kendaraan canggih lainnya, kerangka kerja yang diusulkan dapat dengan mudah diperluas dan diterapkan pada deteksi dan pengenalan jenis objek lain yang ditemui di lingkungan mengemudi, seperti plat nomor, pejalan kaki, rambu lalu lintas, dan sebagainya. Performa yang baik dari algoritma yang diusulkan pada deteksi kendaraan memiliki nilai referensi yang tinggi di bidang mengemudi cerdas. Kedepannya, makalah ini akan menginvestigasi lebih banyak perangkat tambahan untuk meningkatkan hasil deteksi.

Pada penelitian yang dilakukan oleh Ravalisri, dkk pada tahun 2019 dengan judul "*Instance Segmentation on Real time Object Detection using Mask R-CNN*", *Mask* R-CNN menggunakan *Regional Proposal Networks* (RPN) untuk mengidentifikasi objek yang diminati [13]. Dalam penelitian ini, kami telah menjelaskan bahwa dengan *Mask* R-CNN untuk segmentasi *Instance*, kecepatan pembelajaran jauh lebih rendah dibandingkan dengan segmentasi semantik. Selain

itu, kita dapat dengan mudah mengubah ukuran gambar karena kita memiliki rangkaian jaringan saraf konvolusional sendiri untuk memproses gambar masukan. Segmentasi instan dapat digunakan di berbagai bidang dan teknologi seperti mobil *self-driving real-time*. Dimana mobil dapat mendeteksi keberadaan benda di depannya dan melakukan pergerakan yang aman sehingga tidak terjadi kerusakan pada kendaraan, orang lain dan benda disekitarnya. Segmentasi *instance* ini digunakan di berbagai bidang seperti deteksi wajah, *real-time* orang menghitung atau menghitung objek dalam gambar, deteksi fitur pada gambar, seperti mendeteksi sel kanker pada gambar, dapat digunakan dalam bingkai lalu lintas untuk menentukan kendaraan yang dibutuhkan. Implementasi segmentasi *instance* menggunakan *Mask* R-CNN dalam deteksi objek secara *real-time* telah mendapatkan akurasi yang lebih tinggi dibandingkan dengan metode R-CNN sebelumnya.

Pada penelitian yang dilakukan oleh Vinh Dinh Nguyen, dkk pada tahun 2020 dengan judul "*Robust Vehicle Detection Under Adverse Weather Conditions Using Auto-encoder Feature*", yang meneliti tentang investigasi fitur *auto-encoder* untuk meningkatkan kinerja pendeteksian kendaraan yang ada pada kondisi cuaca buruk [14]. Makalah ini menyajikan metode pendeteksian kendaraan yang kuat menggunakan *deep learning auto-encoder* tanpa pengawasan yang diusulkan. Sistem yang diusulkan meningkatkan kinerja metode deteksi hambatan berbasis *deep learning*, seperti *Faster* R-CNN dan YOLO dalam kondisi mengemudi yang sulit. Kerangka kerja yang diusulkan fleksibel, mudah diterapkan untuk meningkatkan akurasi deteksi objek berbasis *deep learning* lainnya. Hasil penelitian ini diperoleh akurasi 96,86%, 93,85%, 84,62%, 80,44% , dan 95,10% masing-masing dalam kondisi malah hari.

Pada penelitian yang dilakukan oleh Mahmoed Muthana, dkk pada tahun 2021 dengan judul "*Dual Architecture Deep learning Based Object Detection System for Autonomous Driving*", yang meneliti tentang R-CNN yang lebih cepat dengan model FPN melaporkan persimpangan yang lebih tinggi di atas *Union* (IoU) dan presisi rata-rata (mAP) daripada YOLOv3 [15]. Sebuah metode disajikan dan diimplementasikan untuk menggunakan arsitektur *multi*-CNN untuk deteksi objek bermata untuk deteksi latar depan dan latar belakang yang lebih baik dengan objek

kecil, terinspirasi oleh faktor fusi yang memengaruhi kinerja deteksi objek kecil. Pendekatan mengusulkan kotak pembatas dengan membandingkan hasil gabungan dari dua arsitektur deteksi objek. Detektornya cepat dan berkinerja lebih baik daripada metode yang disajikan untuk mencapai 52 fps pada GPU. Ini mengungguli pendekatan bermata lain dalam akurasi dan memberikan akurasi yang lebih tinggi pada set data deteksi objek KITTI yang tersedia secara luas. Dalam pekerjaan di masa mendatang, akan memungkinkan untuk bereksperimen dengan jaringan dan algoritme yang berbeda untuk meningkatkan hasil dengan menyetel detektor yang berbeda, tulang punggung yang berbeda, atau kumpulan data yang berbeda. Demikian pula, aplikasi selain deteksi objek dari model yang diusulkan dapat diterapkan dengan modifikasi yang sesuai.

Pada penelitian yang dilakukan oleh Sai Shilpa Padmanabula, dkk pada tahun 2020 dengan judul "*Object Detection Using Stacked* YOLOv3" meneliti tentang model deteksi objek berbasis *Convolutional Neural Network* (CNN) yang canggih dan melatihnya untuk beberapa kelas kendaraan menggunakan data dari jalan Delhi [16]. Kami menerapkan YOLO, model deteksi objek terpadu. Kami memiliki pendekatan deteksi objek yang berbeda seperti R-CNN, R-CNN cepat, R-CNN lebih cepat, YOLO (*You Only Look Once*) dan SSD (*Single shot detector*). R-CNN sangat disengaja. Dibandingkan dengan R-CNN, *faster* R-CNN cepat, tetapi menggunakan pencarian selektif, yang memperlambat proses penemuan. R-CNN yang lebih cepat menggunakan jaringan konvolusional yang disebut jaringan regional alih-alih pencarian selektif, yang membuatnya 10 kali lebih cepat dari R-CNN cepat. Memilih pendekatan yang tepat untuk deteksi objek tergantung pada tugas yang kita selesaikan. Jika akurasi tidak terlalu penting, tetapi kecepatan penting, maka kami menggunakan YOLO untuk tugas komputasi. SSD adalah pilihan terbaik. Berbeda dengan pendekatan berbasis pengklasifikasi, YOLO belajar dari fungsi kerugian dan memiliki hubungan yang buruk dengan kinerja pendeteksian, dan jadi seluruh model dilatih bersama. YOLOv3 adalah detektor objek tujuan umum tercepat dalam literatur, dan YOLO menyelami seni terbaru dari deteksi objek *real-time*. *Stacked* YOLOv3 juga menggeneralisasi dengan baik ke domain baru, menjadikannya ideal untuk aplikasi yang membutuhkan deteksi objek

yang cepat dan andal. Konsep *multi*-skala akan diintegrasikan ke dalam YOLOv3 untuk mengimplementasikan tugas penskalaan dan mendeteksi objek kecil.

Tabel 2.1 Ringkasan Kajian Pustaka

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
1	***A Multi-Scale Deep learning Network For Vehicle Detection***	Hoanh Nguyen	2019	Untuk meningkatkan kinerja pendeteksian kendaraan kecil.	Metode yang diusulkan yaitu *deep learning multi*-skala, dimana peneliti merancang modul yang ditingkatkan yang memadukan lapisan konvolusi yang berbeda dengan menggunakan operasi dekonvolusi dan normalisasi. Dan membandingkan nya dengan metode SSD, *Faster* R-CNN, MS-CNN, dan	Hasil eksperimen pada dataset KITTI menunjukkan keunggulan metode yang diusulkan untuk mendeteksi kendaraan kecil, dan mencapai kinerja yang hampir sama dengan metode canggih lainnya.	Dari perbandingan tersebut hasil terbaik yaitu mendeteksi menggunakan MS-CNN. Namun, MS-CNN lebih lambat dibandingkan dengan metode yang diusulkan oleh penulis, dimana kinerja metode yang diusulkan hampir sama dengan MS-CNN sekaligus lebih cepat dan sederhana	Lebih fokus untuk meningkatkan akurasi

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
					Proposed method.			
2	***Detection of Imaged Objects with Estimated Scales***	Xuesong Li, Ngaiming Kwok, Jose E. Guivant, Karan Narula, Ruowei Li and Hongkun Wu	2019	Mengusulkan metode deteksi berdasarkan perkiraan ukuran objek, melakukan percobaan besar-besaran pada KITTI dataset.	Penelitian ini dilakukan dengan membandingkan 70 *anchor Faster R-CNN* dan 9 *anchor Faster R-CNN*	*Faster* R-CNN dengan 70 *anchor* dievaluasi pada kumpulan data uji dan dibandingkan dengan metode *Faster* R-CNN dengan arsitektur jaringan VGG-16.	Peningkatan akurasi peta adalah 6,66% dan 0,68% dicapai untuk pengendara sepeda dan pejalan kaki, masing-masing	Untuk dapat memperkirakan ukuran *anchor* dan kontinu dari satu gambar.
3	***Improving Faster R-CNN Framework for Fast***	Hoanh Nguyen	2019	Penelitian ini menggunakan kerangka kerja yang diusulkan dapat dengan mudah diperluas dan diterapkan pada deteksi dan	Metode berfokus pada memodifikasi jaringan dasar agar sesuai dengan skala	Performa yang baik dari algoritma yang diusulkan pada deteksi kendaraan memiliki nilai referensi yang	Pada penelitian ini arsitektur MobileNet diadopsi untuk membangun pengklasifikasi	Penelitian ini perlu menginvestigasi lebih banyak perangkat tambahan untuk

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
	Vehicle Detection			pengenalan jenis objek lain yang ditemui di lingkungan mengemudi, seperti plat nomor, pejalan kaki, rambu lalu lintas, dan sebagainya.	yang berbeda dengan menerapkan peta fitur multiskala CNN.	tinggi di bidang mengemudi cerdas.	pada tahap akhir kerangka *Faster* R-CNN untuk mengklasifikasikan proposal dan menyesuaikan kotak pembatas untuk setiap proposal. Pendekatan yang diusulkan dievaluasi pada kumpulan data KITTI dan kumpulan data LSVH. Dibandingkan	meningkatkan hasil deteksi.

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
							dengan kerangka *Faster* R-CNN asli, pendekatan yang diusulkan menunjukkan hasil yang lebih baik dalam akurasi deteksi dan waktu pemrosesan. Hasilnya menunjukkan bahwa jaringan yang diusulkan sederhana, cepat, dan efisien.	

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
4	***Instance Segmentation on Real time Object Detection using Mask R-CNN***	Ravalisri Vasam, Padmalaya Nayak	2019	Apakah Mask R-CNN dapat mencapai hasil tercanggih pada berbagai tugas deteksi objek.	Penelitian ini menggunakan Segmentasi instan dapat digunakan di berbagai bidang dan teknologi seperti mobil *self-driving real-time*. Dimana mobil dapat mendeteksi keberadaan benda di depannya dan melakukan pergerakan yang aman sehingga tidak terjadi kerusakan pada kendaraan, orang lain dan benda disekitarnya.	Segmentasi *instance* ini digunakan di berbagai bidang seperti deteksi wajah, *real-time* orang menghitung atau menghitung objek dalam gambar, deteksi fitur pada gambar, seperti mendeteksi sel kanker pada gambar, dapat digunakan dalam bingkai lalu lintas untuk menentukan kendaraan yang dibutuhkan	Implementasi segmentasi *instance* menggunakan *Mask* R-CNN dalam deteksi objek secara *real-time* telah mendapatkan akurasi yang lebih tinggi dibandingkan dengan metode R-CNN sebelumnya.	Lebih baik, teknik *pre-processing* citra dan transformasi morfologi digunakan untuk mengurangi *noise* dan meningkatkan kejernihan piksel.

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
5	***Robust Vehicle Detection Under Adverse Weather Conditions Using Auto-encoder Feature***	Vinh Dinh Nguyen, Duy Dinh Tran, Man Minh Tran, Nhan Minh Nguyen, Vu Cong Nguyen	2020	Mengevaluasi kinerja sistem yang diusulkan menggunakan tingkat oklusi dan pemotongan yang mudah (seperti yang dijelaskan dalam tolok ukur KITTI).	Penelitian ini menyajikan metode pendeteksian kendaraan yang kuat menggunakan *deep learning auto-encoder* tanpa pengawasan yang diusulkan.	Sistem yang diusulkan meningkatkan kinerja metode deteksi hambatan berbasis pembelajaran mendalam yang ada, seperti Faster RCNN dan YOLO dalam kondisi mengemudi yang sulit. Kerangka kerja yang diusulkan fleksibel, mudah diterapkan untuk meningkatkan akurasi deteksi objek berbasis *deep learning* lainnya.	Hasil penelitian ini diperoleh akurasi 96,86%, 93,85%, 84,62%, 80,44% , dan 95,10% masing-masing dalam kondisi malam hari.	Penelitian ini tidak berfokus pada peningkatan akurasi dalam berbagai situasi tertutup, yang merupakan topik menantang lainnya dalam pendeteksian kendaraan. Jadi dapat dilanjutkan untuk lebih

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
								fokus ke tingkat akurasi
6	***Dual Architecture Deep learning Based Object Detection System for Autonomous Driving***	Mahmoud M. Mahmoud, Ahmed R. Nasser	2021	Pendekatan mengusulkan kotak pembatas dengan membandingkan hasil gabungan dari dua arsitektur deteksi objek	Menggunakan metode klasifikasi citra dan deteksi objek didasarkan pada penggunaan CNN *deep learning*.	Detektornya cepat dan berkinerja lebih baik daripada metode yang disajikan untuk mencapai 52 fps pada GPU. Ini memberikan akurasi yang lebih tinggi daripada dataset deteksi objek KITTI yang tersedia.	Arsitektur multi-CNN untuk deteksi objek, mempengaruhi kinerja deteksi objek kecil. Dan menghasilkan pendeteksian yang mencapai akurasi lebih baik dari dataset deteksi objek KITTI.	Dapat membuat aplikasi selain deteksi objek dari model yang diusulkan dan dapat diterapkan dengan modifikasi yang sesuai.
7	***Vehicle type detection***	Li Suhaoa, Lin Jinzhaoa,	2018	Penelitian ini mengubah	Dalam penelitian ini, model ZF	Hasil percobaan menunjukkan	Hasil percobaan menunjukkan	Dari penelitian ini diharapkan dapat

No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
	based on deep learning in traffic scene	Li Guoquana, Bai Tonga, Wang Huiqiana, Pang Yu		masalah deteksi target menjadi masalah deteksi klasifikasi (termasuk berbagai jenis target kendaraan).	(Zeiler dan Fergus) dan model VGG16 dari algoritma *Faster* R-CNN digunakan untuk melatih dan menyempurnakan parameter dalam model jaringan.	bahwa model jaringan kedalaman (VGG16) mendeteksi lebih banyak jenis target kendaraan yang berbeda dengan lebih akurat.	bahwa dibandingkan dengan metode pembelajaran mesin tradisional, model yang digunakan dalam makalah ini telah ditingkatkan baik dalam akurasi deteksi target rata-rata maupun tingkat deteksi. Hasil pengujian klasifikasi artikel ini juga cocok untuk	dilanjutkan dengan merancang model yang ditingkatkan dan menggunakan data yang dibuat sendiri.

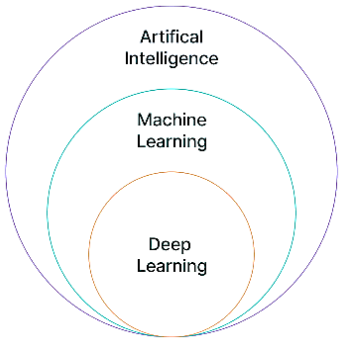
No	*Judul*	*Author*	Tahun	Rumusan Masalah	Metode	Hasil/Temuan	Kesimpulan	Saran
							pendeteksian jenis kendaraan dari tiga jenis mobil, minibus dan suv dalam skenario yang berbeda dan telah mencapai hasil yang baik.	

Perbedaan penelitian terdahulu dengan penelitian ini terletak pada tingkat akurasi, dimana penelitian ini menggunakan arsitektur CNN U-Net. Dengan menggunakan model yang telah dimodifikasi dari dataset KITTI agar menghasilkan tingkat akurasi yang tinggi.

2.2 DASAR TEORI

2.2.1 *Deep learning*

Deep learning merupakan suatu model jaringan syaraf tiruan yang memiliki tingkat akurasi yang tinggi. Selain itu, *deep learning* merupakan cabang dari *Artificial Intelligence* (AI) dan pengembangan dari *Artificial Neural Network* (ANN) namun dengan jaringan yang lebih kompleks [17].



Gambar 2.1 Hubungan AI dengan deep learning [17]

Deep learning untuk saat ini banyak digunakan dan dikembangkan pada kasus pengenalan citra, dikarenakan perkembangan dari *Graphic Processing Unit* (GPU) [18]. Metode *deep learning* disebut juga dengan metode *representation learning* yang mempunyai beberapa tingkat representasi. *Deep learning* menggunakan algoritma *backpropagation* untuk menentukan struktur yang rumit di dalam sebuah dataset yang sangat besar. Kemajuan *deep learning* semakin berkembang pesat dan banyak digunakan dalam komunitas riset dan industri untuk membantu mengatasi berbagai permasalahan data besar, seperti *Computer Vision*, *Speech recognition*, dan *Natural Language Processing* (NLP) [19].

2.2.2 Jaringan Syaraf Tiruan (*Neural Network*)

Jaringan syaraf tiruan, juga dikenal sebagai *Neural Network*, merujuk pada rangkaian tiruan dari sel syaraf atau neuron yang menghubungkan lapisan input dengan lapisan *output*. Ini adalah pendekatan pemodelan yang sangat canggih dan mampu menggambarkan fungsi yang kompleks. Dengan menggunakan jaringan syaraf tiruan, kita dapat melakukan prediksi masa depan, mengklasifikasikan data ke dalam kelompok yang memungkinkan pengambilan keputusan oleh mesin cerdas. Sama seperti manusia, jaringan syaraf tiruan menjalani pelatihan di bawah pengawasan sesuai dengan pendekatan pembelajaran terawasi [20].

Jaringan syaraf tiruan merupakan komponen utama dalam bidang kecerdasan buatan dan pembelajaran mesin. Dalam era pertumbuhan data yang pesat, pembelajaran dalam jaringan ini semakin maju. Oleh karena itu, jaringan syaraf dapat mengatasi peningkatan ukuran dan volume data, serta mengelola aliran informasi yang besar. Salah satu algoritma yang terkenal dengan kinerja akuratnya adalah algoritma *backpropagation*.

Algoritma *Backpropagation*, yang juga dikenal sebagai Algoritma Perambatan Mundur, merupakan sebuah metode pembelajaran yang terarah dan terawasi pada jaringan syaraf tiruan. Tujuannya adalah untuk mencari nilai optimal dari bobot atau beban yang memberikan keunggulan tertentu. Dikarenakan proses pembelajarannya yang berulang-ulang, algoritma ini memiliki kemampuan untuk membangun sistem yang tahan terhadap gangguan dan konsisten dalam kinerjanya [21]. Sehingga, hasilnya adalah nilai kesalahan seminimal mungkin berdasarkan data pembelajaran yang diberikan.

Jaringan Syaraf Tiruan (JST), yang dikenal juga sebagai *Artificial Neural Networks* (ANN), merupakan metode cerdas dalam komputasi canggih yang melakukan analisis kuantitatif terhadap informasi melalui pembelajaran dan pelatihan. Jaringan ini sangat cocok untuk mengatasi model hubungan internal yang kompleks karena sifatnya yang sangat nonlinier. JST juga mampu melakukan pemrosesan paralel pada jumlah data yang besar, memiliki ketahanan tinggi, dan toleransi kesalahan. Jaringan syaraf dapat dikelompokkan menjadi jaringan syaraf *feedforward* (jaringan saraf maju). Dalam jenis jaringan ini, informasi mengalir secara langsung dari lapisan *input* ke lapisan *output* melalui beberapa lapisan tersembunyi (tipe pertama).

Artificial Neural Networks (ANN) merupakan suatu algoritma yang digunakan untuk membentuk pola atau model dari data yang telah ditentukan. Dalam kerangka *Machine Learning* (ML), ANN memiliki kemampuan untuk meningkatkan kinerja sistem dengan memanfaatkan data historis atau sampel data [22]. Banyak penelitian dan pengembangan yang telah dilakukan di dalam bidang kecerdasan buatan, dan salah satu perkembangan yang berasal dari algoritma ANN adalah *Convolutional Neural Network* (CNN). Secara khusus, CNN masuk ke dalam wilayah *Deep Learning* (DL), yang merupakan sub-bidang dari ML. CNN

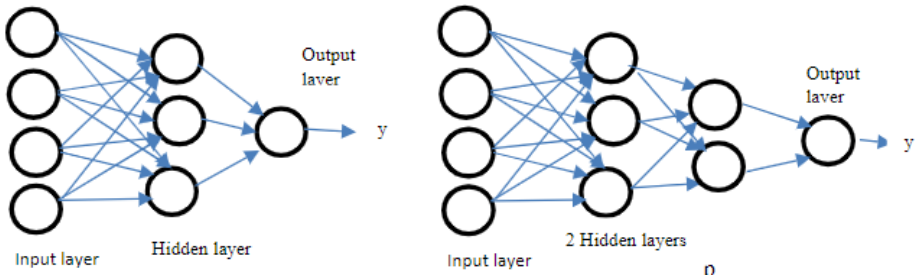
mengambil konsep dasar dari algoritma ANN dan melengkapinya dengan lapisan-lapisan yang lebih kompleks [23].

A. Struktur Jaringan Syaraf Tiruan (*Neural Network*)

Jaringan Syaraf Tiruan (*Neural Network*) terdiri dari tiga tingkatan yang berbeda:

1. Lapisan masukan atau *input* (*buffer*) memiliki beberapa unit masukan neuron (*perceptron*), jumlahnya dapat bervariasi berdasarkan kebutuhan dataset dalam tahap pembelajaran. Neuron dalam lapisan masukan ini tidak memanfaatkan fungsi transfer, tetapi setiap input memiliki faktor skala yang diterapkan untuk mengnormalisasi sinyal. Sinyal yang telah dihitung kemudian diteruskan dari lapisan masukan ke lapisan tersembunyi dan selanjutnya ke lapisan keluaran.
2. Lapisan keluaran atau *output* (*buffer*) terdiri dari beberapa unit neuron keluaran (atau bahkan hanya satu neuron) yang mewakili kelas-kelas dalam kumpulan data keluaran.
3. Lapisan *Hidden* (Tersembunyi) bertindak sebagai penghubung antara lapisan masukan dan keluaran. Informasi mengalir melalui bobot yang dihitung dari lapisan masukan menuju lapisan keluaran. Ketika terjadi kesalahan, sinyal dikirimkan kembali ke lapisan masukan melalui proses propagasi balik (umpan balik). Lapisan tersembunyi juga terdiri dari berbagai unit neuron tersembunyi [24].

Jaringan syaraf dapat memiliki satu atau lebih lapisan tersembunyi, seperti yang digambarkan dalam Gambar 2.2. Setiap neuron dalam lapisan masukan terhubung dengan neuron-neuron dalam lapisan tersembunyi. Meskipun, bagaimana tepatnya komputasi data diproses dalam lapisan tersembunyi tetap sangat kompleks dan belum sepenuhnya dipahami.



Gambar 2.2 Struktur Jaringan Syaraf [24]

Dari gambar diatas terdapat dua jenis, dimana gambar sebelah kiri terdiri dari satu lapisan *input*, satu lapisan *hidden* (tersembunyi), dan satu lapisan *output*. Dan gambar sebelah kanan memiliki satu lapisan *input*, dua lapisan tersembunyi, dan satu lapisan *output*.

Jaringan saraf menghitung *output y* dengan menjumlahkan perkalian vektor bobot (w_i) dan vektor masukan (x_i), serta dengan menggunakan ambang batas yang telah ditentukan (t), untuk menentukan keluaran atau nilai prediksi (y). Berikut merupakan rumus matematisnya.

$$y = \begin{cases} 1 \; if \; \sum_i w_i x_i > t, \\ 0 \; if \; \sum_i w_i x_i \leq t \end{cases} \tag{2.1}$$

Kondisi ambang batas digantikan dengan konstanta yang disebut bias (b), yang setara dengan (-ambang batas, t). Dapat dilihat sebagai berikut.

$$y = \begin{cases} 1 \; if \; \sum_i w_i x_i + b > 0, \\ 0 \; if \; \sum_i w_i x_i + b \leq 0 \end{cases} \tag{2.2}$$

Ketika nilai bias $b > 0$, maka *output*nya adalah 1, dan ketika $b \leq 0$, maka *output*, y adalah 0. Perubahan kecil pada bobot atau bias akan mengubah *output* dan menghasilkan hasil yang lebih mendekati perilaku yang diinginkan [24].

B. Optimasi Jaringan Syaraf Tiruan

Optimasi Jaringan Syaraf Tiruan (*Neural Network*) mengacu pada proses penyesuaian dan peningkatan kinerja jaringan syaraf tiruan dengan cara mengatur parameter-parameter internalnya. Tujuan utama dari optimasi ini adalah untuk meminimalkan kesalahan atau kerugian (*loss*) dalam tugas yang dilakukan oleh jaringan syaraf tiruan. Penelitian ini menggunakan metode optimasi jaringan syaraf tiruan yaitu *Adaptive Moment Estimation* (Adam).

Algoritma Adam merupakan kombinasi dari Momentum dan RMSProp. Algoritma ini juga menghitung tingkat pembelajaran adaptif untuk setiap parameter. Adam menyimpan rata-rata menurun secara eksponensial dari kuadrat gradien sebelumnya v_t seperti pada AdaDelta dan RMSprop, serta menjaga rata-rata menurun secara eksponensial dari gradien sebelumnya m_t seperti pada Momentum [25]. Persamaan algoritma Adam dijelaskan sebagai berikut:

$$m \leftarrow \beta_1 m - (1 - \beta_1)\nabla_\theta J(\theta) \tag{2.3}$$

$$v_t \leftarrow \beta_2 v_t + \nabla_\theta J(\theta) \otimes \nabla_\theta J(\theta) \tag{2.4}$$

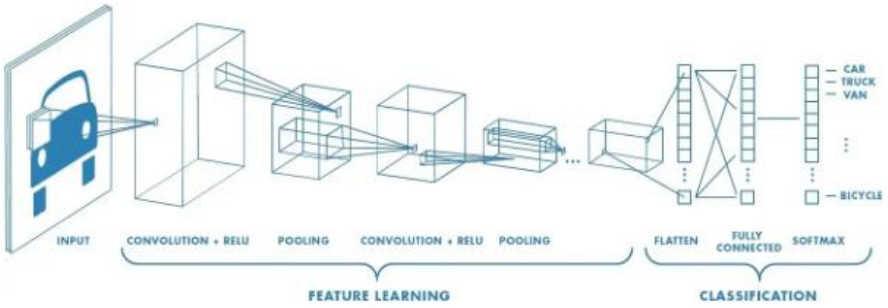
$$m \leftarrow m \otimes (1 - \beta_1^t) \tag{2.5}$$

$$v_t \leftarrow \frac{v_t}{(1-\beta_2^t)} \tag{2.6}$$

$$\theta \leftarrow \theta + \frac{\eta m}{\sqrt{v_t + \mathrm{U}}} \tag{2.7}$$

2.2.3 *Convolutional Neural Network*

Convolutional Neural Network (CNN) merupakan salah satu *Deep learning* untuk *Computer Vision*, yang sering digunakan untuk klasifikasi gambar. CNN digunakan sebagai pendeteksi serta pengenalan objek pada suatu citra. Arsitektur CNN dapat melibatkan beberapa tahap, yang setiap tahapnya terdiri dari tahap *input* dan *output*. Setiap tahap dalam arsitektur ini terdiri dari beberapa *array* yang biasa disebut sebagai *feature map*. Setiap tahap terdiri dari tiga jenis *layer*, yaitu Konvolusi *layer*, Fungsi aktivasi *layer*, *Pooling*. Kombinasi dari tahap-tahap ini membentuk arsitektur dari CNN yang secara keseluruhan dapat melakukan ekstraksi fitur dan pengenalan pola pada citra.



Gambar 2.3 Arsitektur CNN [26]

Dari gambar 2.2 dapat dilihat bahwa arsitektur CNN ada beberapa tahapan. Tahap pertama yaitu tahap konvolusi, pada tahap ini menggunakan sebuah *kernel* dengan ukuran tertentu. Jumlah *kernel* yang digunakan harus dihitung berdasarkan jumlah fitur yang dihasilkan. Selanjutnya yaitu fungsi aktivasi, menggunakan fungsi aktivasi ReLU (*Rectifier Linear Unit*). Setelah tahapan fungsi aktivasi selesai dilanjutkan dengan proses *pooling*. Tahapan tersebut akan diulang beberapa kali sampai mendapatkan peta fitur yang cukup untuk dilanjutkan ke *fully connected neural network* atau *output class* [27].

A. *Convolutional layer*

Convolutional layer merupakan blok utama pada CNN yang digunakan untuk melakukan operasi konvolusi, yang bertujuan sebagai ekstraksi fitur untuk mempelajari representasi fitur dari *input* citra [18]. Pada *layer* ini *input* matriks citra

melakukan operasi dengan matriks-matriks filter, yang nantinya akan menghasilkan keluaran matriks *feature map*. Berikut rumus matematis untuk menghitung ukuran *feature map*.

$$n_{out} = \left(\frac{n_{in}-k+2p}{s}\right) + 1 \quad (2.8)$$

Operasi konvolusi memiliki rumus sebagai berikut.

$$FM[i]_{j,k} = \left(\sum_m \sum_n N_{[j-m,k-n]} F_{[m,n]}\right) + bF \quad (2.9)$$

Dimana :

FM[i] : *Matriks Feature Map* ke-i

j,k : Posisi piksel pada matriks citra masukan

m,n : Posisi piksel pada matriks filter konvolusi

N : Matriks citra masukan

F : Matriks *filter* konvolusi

bF : Nilai bias pada *filter*

Ketika proses konvolusi selesai, kemudian lakukan aktivasi menggunakan fungsi aktivasi ReLU (*Rectified Linear Unit*).

B. *Rectified Linear Unit* (ReLU)

ReLU merupakan fungsi linier yang sering digunakan dalam CNN. ReLU akan mengubah nilai piksel pada *feature map* yang masuk ke dalam fungsi ReLU menjadi 0 untuk mengubah nilai negatif dari *layer* sebelumnya [28]. Persamaan yang digunakan untuk mengubahnya yaitu sebagai berikut.

$$re(x) = \max(0,x) \quad (2.10)$$

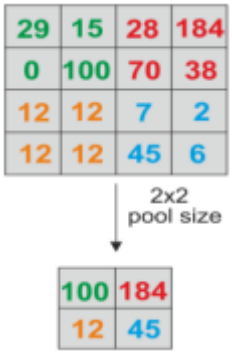
re(x) : Fungsi ReLU dari nilai x

x : *Input*

max(0, x) : Fungsi nilai *max* dari 0 dan x

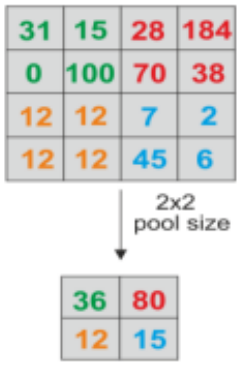
C. *Pooling layer*

Pooling layer merupakan proses mengurangi ukuran *feature map* pada *layer* sebelumnya tanpa merubah informasi pada gambar. Proses *pooling* yang umum digunakan yaitu *max pooling* (*maxpool*), dimana pada setiap area dengan luas piksel tertentu akan diambil nilai tertingginya. Gambar 2.2.3 menunjukkan bahwa saat dilakukan operasi maxpooling (2,2) pada citra berukuran 4x4 piksel, hasilnya akan menjadi citra berukuran 2x2 piksel dengan nilai tertinggi [28].



Gambar 2.4 Proses *Maxpool* [28]

Selain proses *maxpoll* terdapat juga proses *average pooling*, dapat dilihat pada gambar 2.4 dimana *average pooling* akan diambil nilai rata-rata dari semua piksel dalam area piksel yang dipilih.



Gambar 2.5 Proses *Average Pooling* [23]

A. *Fully connected layer*

Fully connected layer adalah sebuah lapisan yang terbuat dari kumpulan hasil proses konvolusi. Suatu citra akan menjadi nilai *input* pada *fully connected layer* setelah melewati proses konvolusi. Proses *fully connected layer* terjadi di bagian akhir arsitektur *Convolutional Neural Network* (CNN). Pada tahap ini, prosesnya direpresentasikan sebagai perkalian antara matriks sederhana dengan *input*, diikuti oleh penambahan vektor bias, dan kemudian menerapkan fungsi *non-linear* berikut [30].

$$y = f(W^T x + b) \tag{2.11}$$

Dimana :

y : Nilai vektor dari *output* yang berisi fungsi dari W

W : Matriks sederhana dengan bobot koneksi antar unit

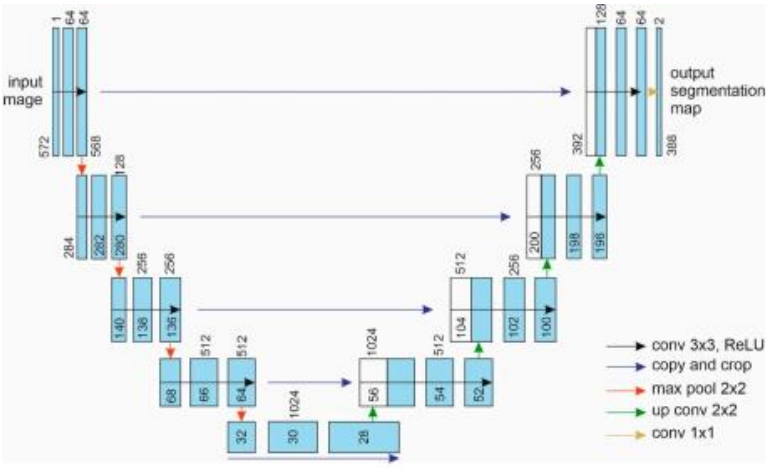
χ : Vektor *input*

b : Vektor bias

2.2.4 Arsitektur U-Net

Jaringan syaraf U-Net dikembangkan pada tahun 2015 untuk pemrosesan gambar medis. Arsitekturnya berasal dari pengembangan dan perbaikan jaringan syaraf konvensional. Masalah defektometri tidak terbatas hanya pada klasifikasi gambar menjadi yang mengandung cacat dan yang tidak. Penting juga untuk menemukan kerusakan dengan akurasi. Selain meningkatkan kontrol kualitas produk, hal ini memungkinkan perhitungan karakteristik kuantitatif kerusakan (luas, arah, dll.). Ini juga memungkinkan untuk lebih memahami sifat kerusakan dan mengembangkan langkah-langkah untuk menghilangkannya. Untuk mencapai hal ini, jaringan syaraf U-Net menyediakan segmentasi semantik gambar, di mana setiap piksel gambar diklasifikasikan sebagai milik salah satu kelas kerusakan atau daerah yang tidak rusak. Dalam hal ini, gambar masukan dan keluaran memiliki ukuran yang sama.

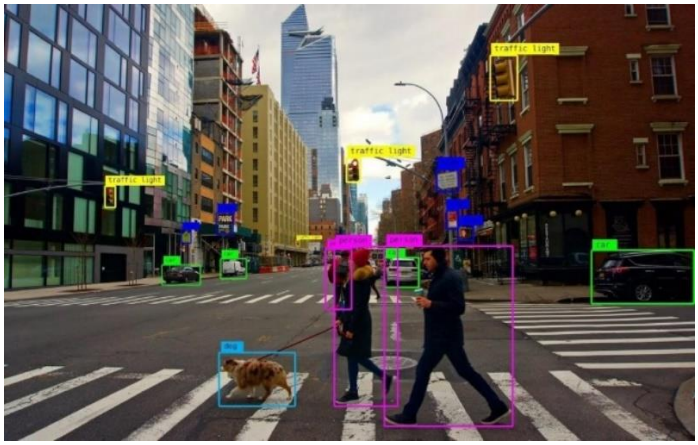
Arsitektur umum dari jaringan syaraf U-Net dasar ditunjukkan dalam Gambar 2.6. Ia simetris dan terdiri dari dua bagian utama: bagian kompresi, yaitu pengode (di sebelah kiri), dan bagian ekspansi, yaitu pemulihan informasi (di sebelah kanan). Bagian kompresi adalah arsitektur tipikal dari jaringan syaraf konvensional, yang mengandung konvolusi berulang dengan kernel 3 × 3, diikuti oleh operasi ReLU (rectified linear unit) dan penggabungan maksimal (max pooling). Dengan setiap pengurangan ukuran gambar, jumlah peta fitur menggandakan. Ketika gambar membesar, ia secara bertahap kembali ke ukuran aslinya. Setiap langkah dari bagian ekspansi meningkatkan dimensi peta fitur, melakukan dekonvolusi, yang mengurangi jumlah peta fitur menjadi setengah, dan menggabungkannya dengan peta fitur yang sesuai dari bagian kompresi. Menggabungkan gambar dengan hasil konvolusi dari lapisan sebelumnya memberikan akurasi yang lebih besar. Pada akhir setiap pengembangan, konvolusi dengan kernel 3 × 3 dan fungsi aktivasi ReLU diterapkan. Akibat dari pengembangan yang diperluas, piksel-piksel baru dimasukkan di antara yang sudah ada, hingga gambar mencapai ukuran yang diinginkan. Lapisan akhir menggunakan konvolusi 1 × 1, yang memproyeksikan setiap vektor dengan fitur ke jumlah kelas yang diinginkan [31].



Gambar 2.6 Arsitektur U-Net [31]

2.2.5 *Computer Vision*

Computer Vision merupakan cabang dari Kecerdasan Buatan (*Artificial Intelegence*/AI), di mana sebuah komputer dilatih agar dapat melihat konten digital berupa citra ataupun video layaknya manusia, yang bertujuan untuk mengekstrak informasi yang terdapat di dalamnya [32]. Ballad dan Brown (1982) mendefinisikan *Computer Vision* sebagai upaya awal dalam mengotomatisasi dan mengintegrasikan pemrosesan serta representasi visual dalam suatu tahapan persepsi visual tertentu.



Gambar 2.7 Contoh Penerapan *Computer Vision* [33]

2.2.6 Citra

Secara harfiah, citra merujuk pada gambar atau representasi dari objek yang ada dalam bidang dua dimensi. Dalam perspektif matematika, citra dapat dianggap sebagai fungsi kontinu dari intensitas cahaya yang berada di bidang dua dimensi. Intensitas cahaya digunakan untuk menerangi objek, dan dari objek tersebut,

sebagian cahaya dipantulkan kembali. Pantulan cahaya ini kemudian dapat ditangkap oleh alat-alat optik seperti mata manusia, kamera, *scanner*, dan sejenisnya, sehingga bayangan objek direkam dalam bentuk citra. Citra dapat dihasilkan sebagai keluaran dari sistem analog berupa sinyal video, seperti gambar yang ditampilkan di layar televisi. Citra juga dapat berupa keluaran dari sistem digital yang dapat langsung disimpan dalam bentuk data pada pita magnetik atau bentuk lainnya. Selain itu, citra dapat dihasilkan melalui sistem perekaman data yang bersifat optik, seperti foto. Dengan demikian, citra merupakan representasi visual dari objek dalam bentuk dua dimensi yang dibentuk melalui proses pemantulan cahaya dan dapat terekam dalam berbagai format dan media. [34].

Sebuah citra terdiri dari beberapa piksel (elemen gambar) yang memiliki koordinat (x,y) dan amplitudo f(x,y). Koordinat (x,y) menunjukkan letak atau posisi piksel pada citra [35]. Citra dapat dikelompokkan menjadi dua jenis, yaitu citra diam (still image) dan citra bergerak (moving image). Citra diam adalah citra tunggal yang tidak mengalami perubahan atau pergerakan, sedangkan citra bergerak terdiri dari rangkaian citra diam yang ditampilkan secara beruntun (sekuensial), sehingga memberikan kesan pada mata manusia sebagai gambar yang bergerak [36].

A. Citra RGB

Citra RGB merupakan citra yang mempunyai tiga komponen warna di setiap pikselnya, yaitu R (*Red*), G (*Green*), dan B (*Blue*). Setiap elemen warnanya memiliki nilai 8 bit, yang artinya nilai tersebut berkisar antara 0 hingga 255. Dari pengertian tersebut dapat memberikan kemungkinan total warna hingga 255 x 255 x 255 atau jika dijumlah menjadi 16.581.375 warna [37].

Gambar 2.8 Representasi RGB [35]

B. Citra Grayscale

Citra *grayscale* (keabuan) merupakan citra yang memiliki nilai piksel kisaran 0 (hitam) sampai 255 (putih), dimana setiap pikselnya berukuran 1 *byte* atau 8 bit. Citra *grayscale* didapatkan dari konversi citra RGB, persamaan yang digunakan untuk konversi nilai RGB menjadi citra *grayscale* dapat dilihat pada persamaan (2.7) [35].

$$Grayscale = 0.299R + 0.5870G + 0.1140B \quad (2.12)$$

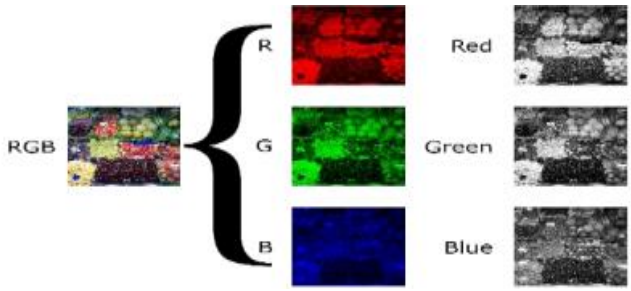
Dimana :

R = Nilai piksel Red (merah)

G = Nilai piksel Green (hijau)

B = Nilai piksel Blue (biru)

Citra hasil konversi dari RGB ke citra grayscale dapat dilihat pada Gambar 2.6.



Gambar 2.9 Proses Konversi RGB ke *Grayscale* [35]

C. Citra Biner

Citra biner merupakan citra dengan nilai setiap pikselnya hanya dinyatakan dengan dua kemungkinan yaitu 0 (hitam) dan 1 (putih). Untuk mendapatkan citra biner dapat melalui konversi citra *grayscale* dengan menggunakan operasi *thresholding*. Operasi *thresholding* umumnya digunakan dalam proses segmentasi citra, di mana tindakan ini membagi nilai derajat keabuan pada setiap piksel menjadi dua kelas, yaitu hitam dan putih. Selain itu, operasi ini digunakan untuk memisahkan objek yang diinginkan (*foreground*) dari objek yang tidak diinginkan (*background*). Ketika hasil operasi melebihi nilai *threshold* yang telah ditentukan, objek *foreground* direpresentasikan oleh nilai 1 (putih). Sebaliknya, jika hasil operasi kurang dari nilai *threshold*, objek *background* direpresentasikan oleh nilai 0 (hitam) [35]. Berikut merupakan persamaan berdasarkan nilai *threshold* (T).

$$g(x,y) = \begin{cases} 1\ jika\ f(x,y) \geq T \\ 0\ jika\ f(x,y) < T \end{cases} \quad (2.13)$$

Dimana :

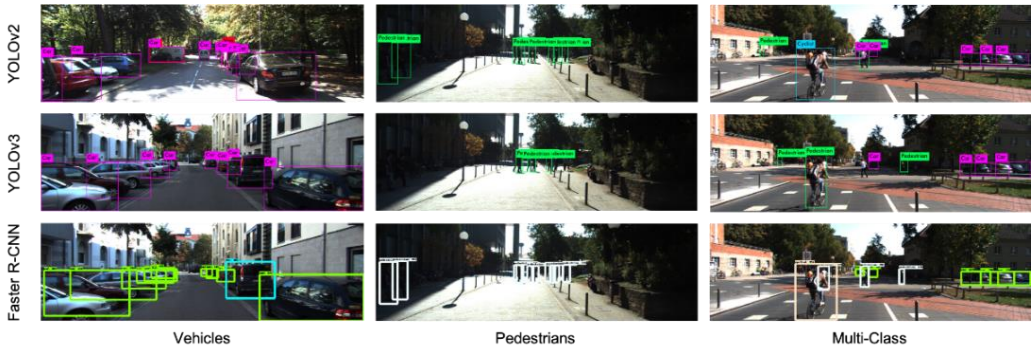
g(x,y) = Citra Biner

f(x,y) = Citra *Grayscale*

T = Nilai *Threshold*

2.2.7 *Object Detection*

Object detection adalah sebuah teknologi komputer yang bertujuan untuk mengidentifikasi dan menentukan lokasi dari objek-objek yang telah dikenal kategorinya (*class*) dalam citra digital yang diam (statis) atau *frame* video [38]. Terdapat beberapa kasus yang menerapkan teknik *object detection* diantaranya yaitu untuk deteksi wajah, deteksi kendaraan, aplikasi untuk mendukung kota cerdas (*smart city*), dan deteksi pemain dan bola pada video siaran sepak bola [39].



Gambar 2.10 *Object Detection* dengan Beberapa Metode [33]

2.2.8 *Model Evaluation*

A. IoU (*Intersection over Union*)

IoU adalah metrik evaluasi yang umum digunakan dalam tugas segmentasi dan deteksi objek. Metrik ini mengukur sejauh mana tumpang tindih antara area yang diprediksi oleh model dan area sebenarnya dari objek yang ada. IoU berguna untuk mengukur seberapa baik model segmentasi atau deteksi objek dapat mengatasi tumpang tindih dan posisi yang tepat antara hasil prediksi dan objek sebenarnya.

IoU dihitung sebagai rasio dari luas tumpang tindih antara area prediksi dan area sebenarnya dari objek terhadap luas gabungan dari keduanya. Formula IoU adalah sebagai berikut:

$$IoU = \frac{(Area\ of\ Intersection)}{(Area\ of\ Union)} \tag{2.14}$$

Dimana :

- *Area of Intersection* adalah luas daerah di antara dua objek yang tumpang tindih.
- *Area of Union* adalah luas daerah gabungan dari kedua objek.

Dalam konteks segmentasi atau deteksi objek, IoU mengukur persentase area yang benar-benar diklasifikasikan dengan benar dalam area prediksi. Semakin tinggi nilai IoU, semakin baik model dapat memetakan objek dengan akurat.

IoU sangat berguna dalam mengukur performa model dalam tugas-tugas seperti segmentasi semantik, segmentasi instans, dan deteksi objek, karena memberikan pemahaman tentang sejauh mana model mampu mengenali dan mengatasi tumpang tindih antara objek dalam gambar [41].

B. Performa Model

Performa model atau hasil evaluasi model dalam hal ini bertujuan untuk menggambarkan sejauh mana kemampuan model untuk melakukan prediksi yang akurat dan relevan terhadap data baru yang tidak pernah dilihat sebelumnya. Terdapat tiga kondisi yang terjadi saat melatih model *machine learning* atau *deep learning*.

1. *Overfitting*

Overfitting terjadi ketika model belajar dari data pelatihan terlalu baik sehingga menjadi terlalu spesifik untuk data pelatihan itu sendiri, dan akibatnya tidak dapat melakukan generalisasi dengan baik pada data baru. Penyebab utama *overfitting* adalah model memiliki terlalu banyak parameter yang kompleks atau memiliki kapasitas yang berlebihan. *Overfitting* dapat diatasi dengan teknik-teknik seperti regulasi (misalnya L1/L2 *regularization, dropout*), penggunaan lebih banyak data pelatihan, dan pemilihan model yang lebih sederhana [42].

2. *Underfitting*

Underfitting terjadi ketika model tidak dapat belajar dengan baik dari data pelatihan. Penyebab utama *underfitting* adalah model terlalu sederhana atau memiliki terlalu sedikit parameter. *Underfitting* dapat diatasi dengan menggunakan model yang lebih kompleks, menambahkan lebih banyak fitur atau memperluas fitur yang ada, atau memperluas data pelatihan [42].

3. *Good Fit*

Good Fit adalah kondisi di mana model *machine learning* atau *deep learning* memiliki performa yang baik pada data pelatihan dan juga mampu melakukan

prediksi yang baik pada data baru yang belum pernah dilihat sebelumnya. Model dengan kondisi *Good Fit* cenderung memiliki tingkat akurasi yang baik dan mampu melakukan generalisasi dengan baik pada data baru [42].