

BAB 2

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Pada penelitian ini ada beberapa referensi terkait implementasi algoritma YOLO yang digunakan sebagai acuan, antara lain sebagai berikut:

Menurut Kenneth Dawson & Howe (2014: 1) menjelaskan bahwa *Computer Vision* adalah sebuah analisis gambar dan video secara otomatis oleh komputer guna mendapatkan suatu pemahaman mengenai dunia. *Computer Vision* terinspirasi dari kemampuan sistem penglihatan manusia.

Penelitian Matthijs Hollemans (2015), dengan judul “*Real-Time Object Detection With YOLO*”. Tujuan dari penelitian ini adalah untuk mendeteksi objek secara *real-time* dengan algoritma YOLO, hasil yang didapatkan adalah sistem dapat mengenali berbagai macam objek.

Penelitian dari Junita Sri Wisna H (2020), dengan judul “*Deteksi Kendaraan Secara Real Time Menggunakan Metode YOLO Berbasis Android*”. Tujuan dari penelitian ini adalah untuk melakukan pendeteksian kendaraan di jalan raya dengan menggunakan kamera android sebagai bahan latih sedangkan untuk implementasinya menggunakan kamera pemantau lalu lintas di jalan raya.

Penelitian dari Calvin Geraldy, Chairisni Lubis (2021), dengan judul “*Pendeteksian dan Pengenalan Jenis Mobil Menggunakan*

Algoritma You Only Look Once dan Convolutional Neural Network". Tujuan dari penelitian ini adalah untuk mendeteksi dan mengenali lebih dari 1 jenis mobil dalam gambar / video.

Viktoria Plemakova, (2018), dengan judul "*Vehicle Detection Based on Convolutional Neural Networks*". Tujuan dari penelitian ini adalah untuk mendeteksi kendaraan pada gambar.

Jacob Solawetz, (2020), dengan judul "*How to Train A Custom Object Detection Model with YOLOv5*". Tujuan dari penelitian ini adalah untuk melatih model dengan custom object menggunakan YOLOv5.

Penelitian yang akan dilakukan dengan judul "Implementasi Algoritma YOLO Untuk Mendeteksi Objek Kendaraan Bermotor (Studi kasus : Pintu masuk parkir (UTDI)". Tujuan dari penelitian ini adalah untuk mendeteksi kendaraan dan menghitung jumlah kendaraan yang lewat di pintu masuk parkir UTDI.

Tabel 2.1 Tinjauan Pustaka

Penulis	Studi Kasus	Metode	Objek	Interface
Matthijs Hollemans (2015)	Real Time Detection With YOLO	YOLO	Objek Sekitar Rumah	iOS
Junita Sri Wisna H (2020)	Deteksi Kendaraan Berbasis Android	YOLO	Kendaraan	Android
Calvin Gerald, Chairisni Lubis (2021)	Deteksi dan Pengenalan Jenis Mobil	YOLO, CNN	Mobil	Web
Viktoria Plemakova (2018)	Vehicle Detection on Image	CNN	Kendaraan Pada Gambar	Web
Jacob Solawetz (2020)	Train Custom Object Detection	YOLOv5	Blood image	Web
Ikhsan Syahrizal (Usulan)	Implementasi Algoritma YOLO Untuk Mendeteksi Objek Kendaraan Bermotor	YOLOv3	Motor	Desktop

2.2 Dasar Teori

2.2.1 Deteksi Objek

Menurut kamus besar bahasa Indonesia arti kata deteksi adalah usaha untuk menemukan dan menentukan keberadaan, anggapan, atau kenyataan (KBBI, 2016). Jadi dapat diartikan bahwa deteksi objek adalah usaha untuk menemukan, menentukan keberadaan, anggapan, atau kenyataan suatu objek.

Computer Vision (CV) merupakan kecerdasan buatan untuk melihat, dan mengartikan data visual. Menurut Adrian Low *Computer vision* berhubungan dengan perolehan gambar, pemrosesan, klarifikasi, pengenalan, dan menjadi penggabungan, pengurutan pembuatan keputusan menuju pengenalan (Adrian Low, 1991).

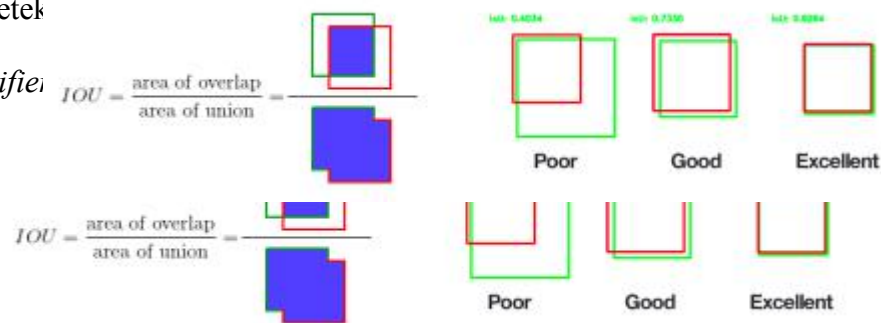
2.2.2 Algoritma YOLO

A. YOLO

Banyak Algoritma yang digunakan untuk pembuatan kecerdasan buatan dalam deteksi objek, pada penelitian ini digunakan algoritma YOLO. Algoritma YOLO (*You Only Look Once*) adalah sebuah algoritma yang dikembangkan untuk mendeteksi sebuah objek secara *real-time*. Sistem pendetek

classifie

pada



Gambar 2.1 Matrik IoU (Cowton, 2019)

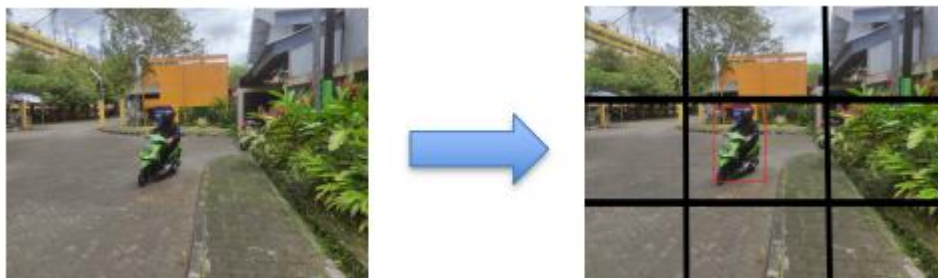
IoU (*Intersection Over Union*) adalah matrik evaluasi yang digunakan untuk mengukur keakuratan detektor objek (Cowton, 2019). IoU dapat mengetahui seberapa baik kotak prediksi kita tumpang tindih dengan kotak sebenarnya, semakin tinggi IoU semakin baik kinerja.

Keuntungan menggunakan algoritma YOLO sendiri yaitu algoritma yang dapat mendeteksi objek super cepat (45 fps) dan akurat. Tidak seperti algoritma yang lain yang menggunakan banyak iterasi sehingga membuat pendeteksian objek menjadi lama, YOLO hanya menggunakan 1 iterasi.

Namun YOLO bukan berarti tidak memiliki kekurangan, permasalahan yang ada pada YOLO sendiri yaitu kesulitan dalam mendeteksi objek yang kecil dan muncul dalam berkelompok karena setiap sisi dibatasi untuk mendeteksi hanya 1 objek seperti contoh barisan semut.

B. Cara Kerja YOLO

Cara kerja YOLO sendiri yaitu dengan membagi gambar menjadi N grid, masing - masing memiliki wilayah dimensi $S \times S$ yang sama.



Gambar 2.2 Contoh objek yang terdeteksi oleh YOLO

YOLO memprediksi kotak pembatas sel per grid, satu kotak prediktor bertanggung jawab untuk memprediksi objek berdasarkan prediksi mana yang memiliki IOU tertinggi.

$y =$	pc
	bx
	by
	bh
	bw
	c1
	c2
	c3

Gambar 2.3 Prediksi YOLO (P Sharma, 2018)

Dimana Gambar 2.3 :

pc = objek

bx = koordinat x tengah objek (0-1)

by = koordinat y tengah objek (0-1)

bh = tinggi *bounding box*

bw = lebar *bounding box*

c1, c2 ,c3 = kelas objek

contoh jika ada dan tidak ada objek terdeteksi pada kotak prediksi :



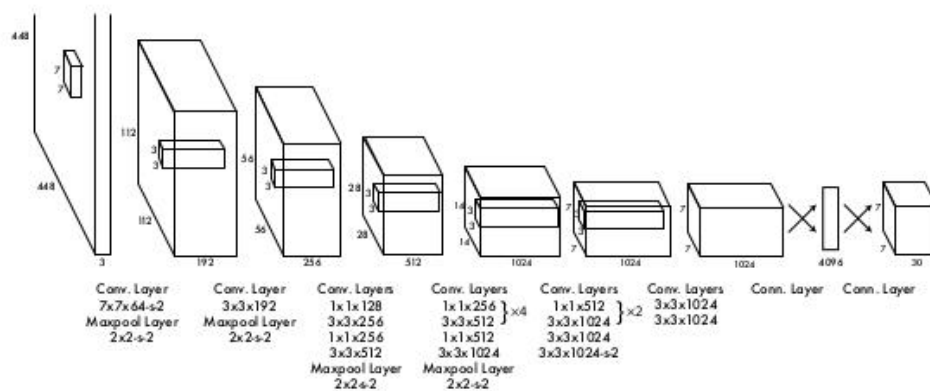
Gambar 2.4 Grid yang tidak mendeteksi objek



Gambar 2.5 Grid yang mendeteksi objek

C. Arsitektur YOLO

Arsitektur YOLO tersendiri terinspirasi dari model GoogLeNet yang digunakan dalam tugas klasifikasi gambar (Joseph Redmon, 2016). Terdiri dari 3 jenis lapisan : Convolutional, Maxpool, dan Fully Connected layer.

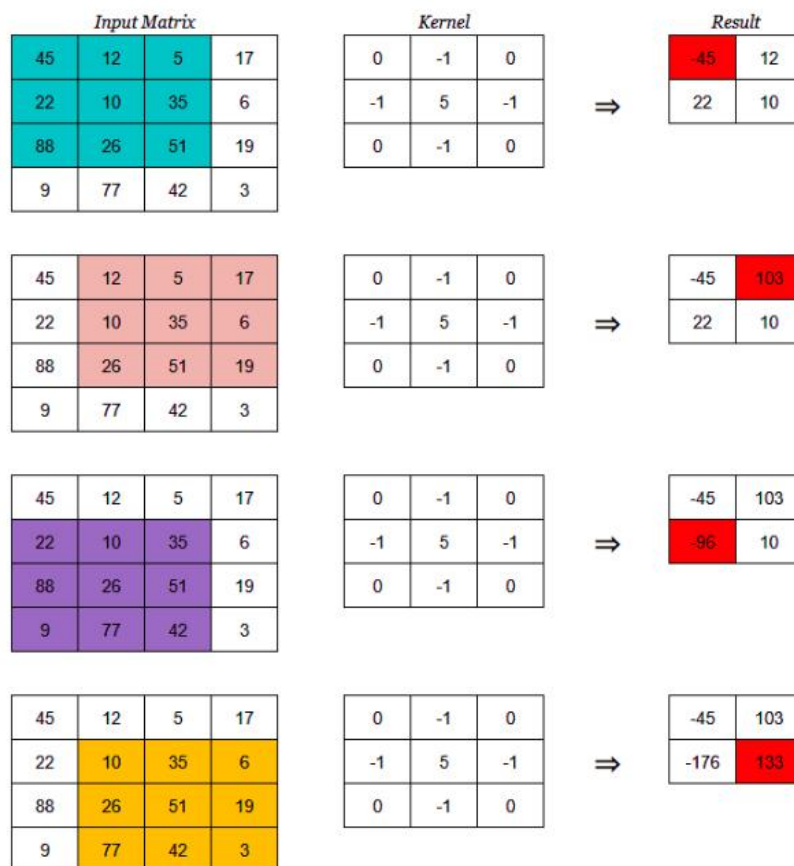


Gambar 2.6 Arsitektur YOLO (Aditya Yanuar, 2018)





Arsitektur ini mengambil gambar sebagai input dan mengubah ukurannya menjadi 448*448 kemudian diteruskan ke jaringan CNN (conv layer). Model ini memiliki 24 lapisan konvolusi, 4 lapisan max-pooling dan SS2 lapisan yang terhubung penuh. Arsitektur ini menggunakan Leaky ReLU sebagai fungsi aktivasinya di seluruh arsitektur kecuali lapisan terakhir yang menggunakan fungsi aktivasi linear (Aditya Yanuar, 2018).
dimana :

a. *Convolutional* (konvolusi)

Dalam konteks *Convolutional Neural Network* (CNN) adalah operasi linier yang melibatkan penggandaan sekumpulan bobot dengan inputan gambar yang diwakili oleh matrik (Albem, 2021). matrik yang mewakili gambar inputan akan dijumlahkan dengan *kernel* untuk mendapatkan bobot nilainya..



Gambar 2.7 Mengekstrak fitur (P Choudari, 2020)

<i>Original</i>	<i>Gaussian Blur</i>	<i>Sharpen</i>	<i>Edge Detection</i>
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
			

Gambar 2.8 Matrik kernel (D Shahrokhian, 2022)

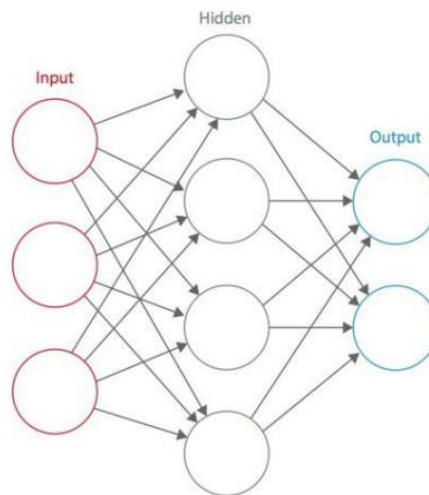
Kernel sendiri merupakan matriks bobot yang dikalikan dengan input untuk mengekstrak fitur yang relevan seperti untuk mempertajam gambar, blur, deteksi tepi dll (D Shahrokhian, 2022)

b. Maxpool

Dikenal sebagai downsampling, melakukan pengurangan dimensi, mengurangi jumlah parameter dalam input, filter akan memilih piksel dengan nilai maksimal lalu dikirim sebagai output (Aini, 2021). Seperti pada gambar 2.7 yang semula input matrik gambar 4x4 di kurangi menjadi 2x2.

c. Fully connected layer

Lapisan ini melakukan tugas klasifikasi berdasarkan fitur yang sudah diekstraksi melalui fitur sebelumnya (Mukherjee, 2019). Menggunakan fungsi linear dengan probabilitas 0 hingga 1. Cara kerjanya adalah dengan menghubungkan setiap *neuron* dalam satu sisi ke setiap *neuron* sisi lainnya, seperti pada gambar 2.9 berikut.



Gambar 2.9 *Fully Connected Layer* (Addison, 2019)

Pada gambar 2.9, terdapat juga proses *flatten* yang berfungsi untuk merubah nilai input menjadi 1 dimensi, kemudian hasil dari proses fully-connected layers akan diperoleh nilai yang dijadikan untuk proses klasifikasi objek yang merupakan nilai output dari hidden-layers sebelumnya.

d. Leaky ReLU

Adalah versi ReLU (Rectified Linear Unit) yang ditingkatkan untuk mengatasi masalah Dying ReLU (neuron mati), dimana jika pada ReLU fungsi 0 jika $x < 0$, pada Leaky ReLU daripada menjadi 0, malah akan memiliki nilai yang kecil (mulai dari 0,01) (S. Singh, 2020). dituliskan dengan persamaan 1 :

$$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x, & \text{for } x \geq 0 \end{cases} \quad (1)$$

dimana persamaan 1 :

$x = 0.01$ (bukan 0)

e. Fungsi aktivasi linear

Dikenal sebagai fungsi identitas / no activation (dikalikan 1,0) adalah dimana aktivasi sebanding dengan input (Yudhi K, 2018). Ini karena fungsi aktivasi linear tidak mengubah jumlah bobot input dengan cara apapun dan sebaliknya mengembalikan nilai secara langsung. Dituliskan dengan persamaan 2 :

$$f(x) = kx \quad (2)$$

dimana persamaan 2 :

k = konstan

D. Dark Net

Dark Net adalah *framework* jaringan saraf tiruan yang bersifat *open source* (bebas / gratis), yang ditulis dalam bahasa C dan CUDA (*Compute Unified Device Architecture*) (Joseph Redmon, 2016). *Framework* ini terkenal dengan algoritma YOLO yang digunakan untuk mendeteksi objek secara *real time*, dan kecepatannya serta kemudahannya untuk di *install*.

E. Euclidean Distance

Euclidean Distance adalah metrik yang ditentukan di atas ruang *Euclidean* (ruang fisik yang mengelilingi kita, plus atau minus beberapa dimensi). Singkatnya, jarak *Euclidean* mengukur jalur terpendek antara dua titik dalam ruang n-dimensi yang mulus (D. Borchia, 2023). Pada penelitian ini digunakan untuk mengukur jarak antara objek untuk

menentukan apakah objek tersebut sama atau tidak, dituliskan dengan persamaan 3 :

$$d(p, q) = \sqrt{\sum_{i=1}^n (q_i - p_i)^2} \quad (3)$$

dimana persamaan 3 :

p, q = dua titik dalam ruang-n Euclidean

q_i, p_i = Vektor Euclidean, mulai dari asal ruang (titik awal)

n = n-ruang

F. Mean Average Precision (mAP)

Mean Average Precision (mAP) adalah metrik yang digunakan untuk mengevaluasi model deteksi objek (Dewa Syah, 2023). Pada penelitian ini digunakan untuk melakukan *testing* model yang telah di-*training* untuk diketahui tingkat akurasi pendeteksiannya, dituliskan dengan persamaan 4 :

$$mAp = \frac{\sum_{q=1}^Q AveP(q)}{Q} \quad (4)$$

dimana persamaan 4 :

Q = jumlah kueri

$AveP(q)$ = presisi rata - rata