BAB I

PENDAHULUAN

A. Latar Belakang

Penglihatan manusia mampu melihat hal-hal yang menakjubkan seperti melihat dan mengenali suatu objek, pandu arah dalam menghadapi rintangan, mengenali suasana hati dalam perilaku, dan mampu membayangkan suatu cerita baik nyata maupun fiksi. Sama halnya dengan mesin, yaitu perangkat komputer. Untuk melakukan penyesuaian layaknya penglihatan manusia, komputer membutuhkan suatu sensor seperti mata manusia dan program komputer yang berfungsi untuk memproses data dari sensor.

Computer vision merupakan ilmu yang memanfaatkan *image processing* untuk membuat sebuah keputusan berdasarkan citra yang didapatkan dari sensor. *Computer vision* mempunyai tujuan untuk membuat perangkat cerdas yang mampu melihat. Salah satu kerangka kerja yang digunakan dalam *computer vision* adalah deteksi objek.

Deteksi objek adalah fenomena dalam *computer vision* yang melibatkan pendeteksian berbagai objek dalam gambar atau video digital. Beberapa objek yang terdeteksi antara lain manusia, hewan, tumbuhan, bangunan, kendaraan, peralatan rumah dll.

Deteksi objek terdiri dari berbagai pendekatan seperti *fast R-CNN, Retina-Net*, dan *Single-Shot Multibox Detector (*SSD). Meskipun pendekatan ini telah

memecahkan tantangan keterbatasan data dan pemodelan dalam deteksi objek, mereka tidak dapat mendeteksi objek dalam menjalankan algoritma tunggal Seperti algoritma YOLO. Alhasil algoritma YOLO mendapatkan popularitas karena kinerjanya yang unggul dibandingkan dengan teknik deteksi objek yang disebutkan diatas.

B. Rumusan Masalah

Berdasarkan uraian diatas maka permasalahan yang timbul dalam pengerjaan proposal penelitian ini bagaimana cara mendeteksi objek pada gambar menggunakan algoritma YOLO?

C. Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk mengetahui cara mendeteksi objek pada gambar menggunakan algoritma YOLO.

D. Batasan Masalah

Untuk menghindari pembahasan yang lebih luas terkait dengan deteksi objek pada gambar menggunakan algoritma YOLO, maka penelitian ini ditentukan pada ruang lingkup tertentu antara lain:

1. Objek yang akan dideteksi hanya berfokus pada gambar
2. Pengumpulan data citra diperoleh secara manual menggunakan *smartphone*

E. Manfaat Penelitian

Adapun manfaat dari hasil penelitian ini diharapkan dapat dijadikan sebagai rujukan dalam mengembangkan deteksi objek menggunakan algoritma YOLO kedepannya pada spesifikasi objek tertentu.

F. SISTEMATIKA PENULISAN

Sistematika yang digunakan dalam penyusunan skripsi ini adalah sebagai berikut:

BAB I PENDAHULUAN

Dalam bab ini akan diuraikan mengenai latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian serta sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Dalam bab ini berisikan beberapa definisi atau pengertian yang mendukung dalam pembahasan penyusunan tugas akhir ini sehingga memudahkan penulis dalam menyelesaikan masalah.

BAB III METODE PENELITIAN

Pada bab ini menjelaskan mengenai tahapan – tahapan yang dilalui dalam penyelesaian tugas akhir ini, yaitu jenis penelitian, lokasi dan waktu penelitian, teknik pengumpulan data, alat dan bahan penelitian serta rancangan sistem yang akan dibuat.

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini menerapkan dan mengimplementasikan tahapan – tahapan yang ada pada bab sebelumnya beserta detail tentang hasil penelitian dalam penyelesaian tugas akhir ini.

BAB V KESIMPULAN

Dalam bab ini menjelaskan mengenai kesimpulan dan saran.

BAB II

TINJAUAN PUSTAKA

A. Penelitian Terdahulu

Penelitian terdahulu pernah dilakukan oleh Mawaddah Harahap, Juni Elfrida, Pasrah Agusman, Mario Rafael, Rahul Abram, dan Kiki Andrianto (2019) berupa jurnal dengan judul "Sistem Cerdas Pemantauan Arus Lalu Lintas Dengan YOLO (You Only Look Once v3)". Penelitian ini menggunakan kerangka kerja YOLO yang dimana dataset diperoleh dari hasil rekaman CCTV pemantauan arus lalu lintas dan kemudian dilakukan pengujian untuk melihat akurasi deteksi objek di setiap persimpangan.

Penelitian yang dilakukan oleh Evan Tanuwijaya dan Chastine Fatichah (2019) berupa jurnal dengan judul "Penandaan Otomatis Tempat Parkir Menggunakan YOLO untuk Mendeteksi Ketersediaan Tempat Parkir Mobil pada Video CCTV" yang mengembangkan sebuah strategi sistem dalam hal deteksi tempat parkir otomatis. Penelitian ini menggunakan strategi yang dibagi menjadi 2 (dua) tahap, yaitu tahap *marking* dimana sistem akan diberi masukan citra tempat parkir yang penuh dengan mobil yang terparkir untuk ditandai. Dan tahap klasifikasi di mana pada proses tempat parkir yang telah ditandai dari proses *marking* akan diklasifikasikan ketersediaan tempat parkir pada lapangan parkir dengan masukan berupa citra atau video dari CCTV.

Penelitian yang dilakukan oleh Muhammad Rafly Tharhan (2020) dengan judul "Rancang Bangun Aplikasi Deteksi Kerusakan Jalan Menggunakan Algoritma Yolo" yang memanfaatkan algoritma YOLO dalam mendeteksi kerusakan pada jalan. Penelitian ini mengimplementasikan aplikasi berbasis android dan pengolahan citra.

Penelitian yang dilakukan oleh Muhammad Harun Alrasyid Nasution (2020) dengan judul "Deteksi Objek Sekitar Menggunakan Algoritma You Only Look Once (YOLO) Untuk Autonomous Electric Vehicle" yang berfokus pada pengujian kinerja algoritma YOLO dalam mendeteksi objek sekitar. Penelitian ini memanfaatkan algoritma YOLO dalam mendeteksi objek sekitar *autonomous vehicle* secara *real-time*.

Penelitian yang dilakukan oleh Muhammad Syarif Hidayatulloh (2021) dengan judul "Sistem Pengenalan Wajah Menggunakan Metode YOLO (You Only Look Once)" yang menggunakan pendekatan metode YOLO dalam mendeteksi wajah, Penelitian ini berfokus pada pengujian dan hasil tingkat akurasi deteksi wajah dengan menggunakan algoritma Yolo.

B. Pengolahan Citra Digital

Pengolahan citra digital memiliki tujuan untuk memperbaiki kualitas suatu citra agar mudah dikenali dan dimanfaatkan oleh manusia dan suatu perangkat yaitu komputer. Teknik dalam pengolahan citra yaitu mengolah suatu citra menjadi citra lain. Sehingga, inputnya adalah citra dan outputnya juga citra, serta

output citra mempunyai kualitas yang lebih baik daripada inputan citra. (Umam & Negara, 2016).

Salah satu *library* pendukung dalam pengembangan pengolahan citra yaitu OpenCV. OpenCV atau *Open Source Computer Vision Library* adalah *open source vision computer* yang dibangun untuk membantu mengelola infrastruktur umum pada aplikasi *computer vision*.

C. Deep Learning

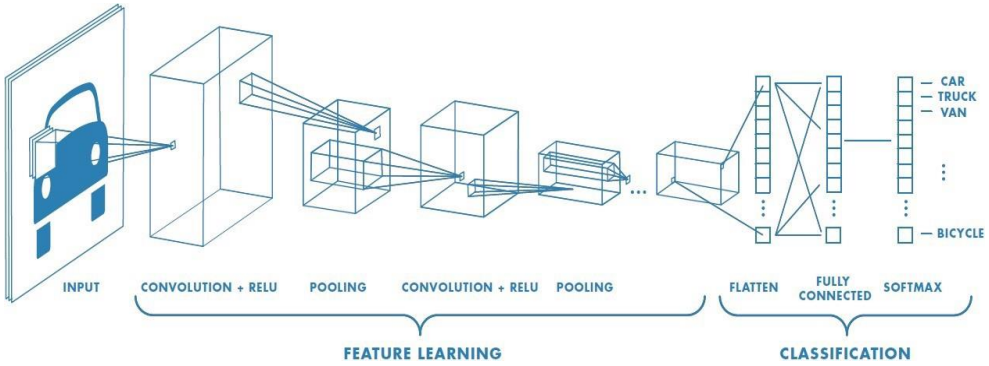
Deep Learning merupakan sebuah bidang *machine learning* dalam bidang keilmuan baru yang berkembang karena perkembangan *GPU acceleration*. *Deep learning* mempunyai kemampuan yang mumpuni dalam *computer vision*, dengan mengimplementasikan metode *machine learning* yang diperlukan untuk mengklasifikasi suatu objek pada citra, yaitu *Convolutional Neural Network* (CNN).

Metode *deep learning* membagi suatu data dalam 2 (dua) sesi, yaitu sesi *training* dan sesi *testing*. Pada sesi *training* mempelajari pemisahan fitur pada setiap data agar mampu membedakan suatu label dengan label yang lainnya. Pada sesi *testing* data dapat dianalisis melalui hasil sesi training.

D. Convolutional Neural Network (CNN)

Convolutional Neural Network adalah jenis *neural network* yang umumnya dimanfaatkan dalam pengolahan data citra. Konvolusi atau *convolution*

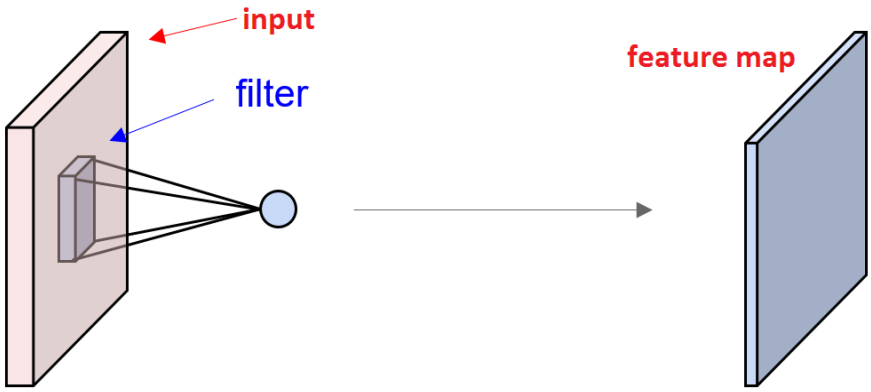
adalah matriks yang mampu menyaring citra. *Convolutional Neural Network* mempunyai beberapa layer untuk menyaring setiap proses yang difungsikan. Prosesnya tersebut disebut dengan proses *training*. Pada proses *training* terdapat 3 (tiga) tahapan utama dalam memproses layer pada suatu citra yaitu *Convolutional Layer*, *Pooling Layer*, dan *Fully Connected Layer*.



Gambar 2. 1 Proses *Convolutional Neural Network*

1. **Convolutional Layer**

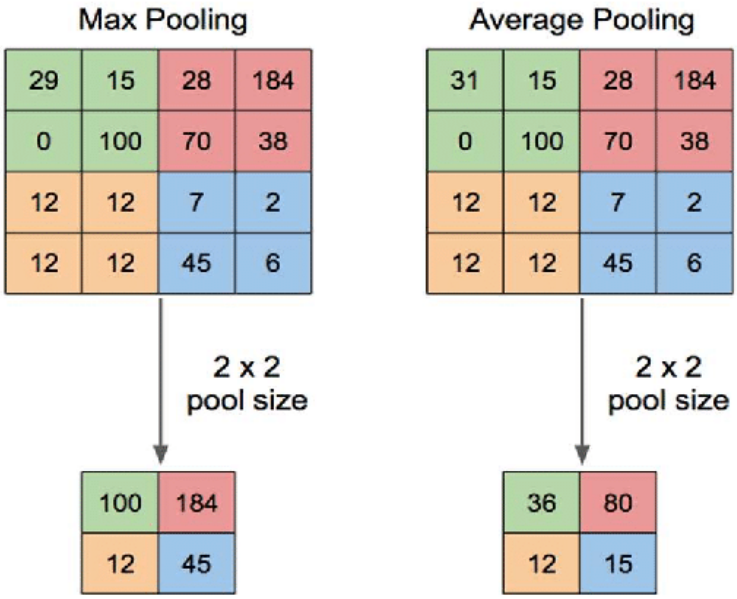
   Setiap data yang menyentuh *Convolutional Layer* akan memasuki proses yang disebut konvolusi. Setiap lapisan akan menyaring inputan data dan menghasilkan *feature map*. Setiap filter yang masuk ke dalam *Convolutional Layer* akan memiliki panjang, tinggi dan tebal. Setiap filter akan mengalami pergeseran antara inputan data dan nilai dari *filter*.



Gambar 2. 2 *Convolutional Layer*

2. Pooling Layer

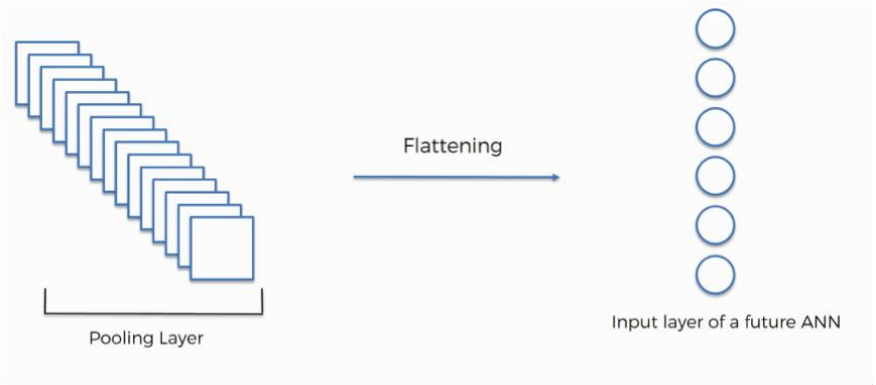
Pooling Layer merupakan tahap setelah *Convolutional Layer*. *Pooling Layer* terdiri dari sebuah *filter* dengan ukuran dan *stride* atau langkah tertentu. Setiap pergeseran akan ditentukan oleh jumlah *stride* atau langkah yang akan digeser pada seluruh area *feature map* atau *activation map*. Dalam *Pooling Layer*, Terdapat 2 (dua) macam *Pooling* yang biasa digunakan yaitu *Max Pooling* dan *Average Pooling*. Misalnya, Jika kita menggunakan *Max Pooling* 2x2 dengan *Stride* 2, maka pada setiap pergeseran *filter*, nilai yang diambil adalah nilai yang terbesar pada area 2x2 tersebut, Sedangkan *Average Pooling* akan mengambil nilai rata-rata.



Gambar 2. 3 *Pooling Layer*

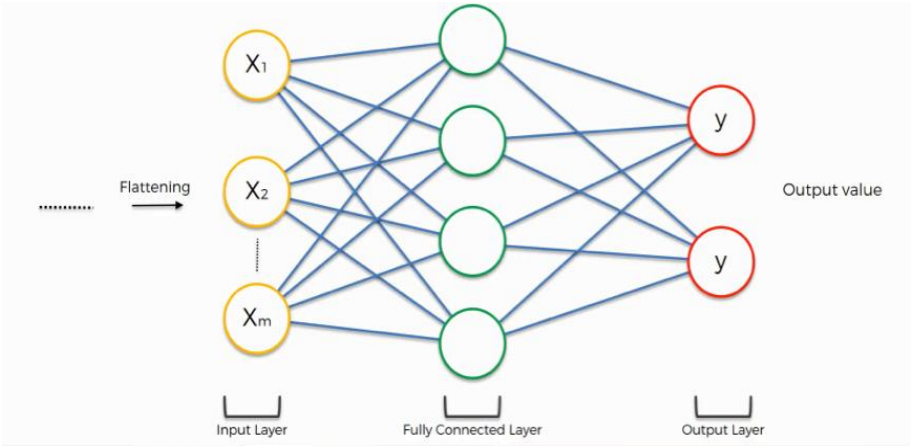
3. Fully Connected Layer

Feature *map* yang diperoleh pada tahap *pooling layer* akan menghasilkan *multidimensional array*. Jadi, sebelum masuk pada tahap *Fully Connected Layer*, *Feature Map* akan masuk ke dalam penyaringan "*flatten*" atau *reshape*.



Gambar 2. 4 *Flatten*

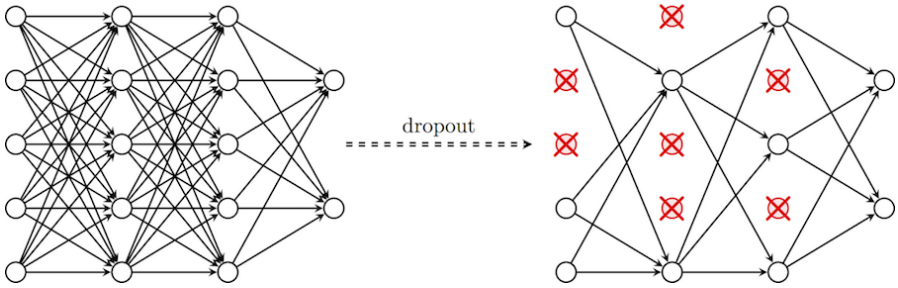
Penyaringan *flatten* akan menghasilkan vektor sebagai input yang digunakan pada *Fully Connected Layer*. *Fully Connected Layer* memiliki beberapa layer diantaranya *Hidden Layer*, *Action Function*, *Output Layer* dan *Loss Function*. Layer ini umumnya digunakan untuk menerapkan *multilayer perceptron* dan memiliki tujuan untuk melakukan perubahan pada dimensi data agar data dapat diklasifikasikan secara linear.



Gambar 2. 5 *Fully Connected Layer*

4. Dropout

Dropout merupakan upaya untuk mencegah terjadinya *overfitting* dan juga mempercepat proses *learning*. *Overfitting* merupakan kondisi dimana hampir semua data yang telah melalui proses *training* mencapai persentase yang baik, Namun terdapat ketidaksesuaian pada proses prediksi. Dalam sistem kerjanya, *Dropout* menghilangkan sementara suatu neuron yang berupa *Hidden Layer* dan *Visible Layer* yang berada di dalam jaringan.



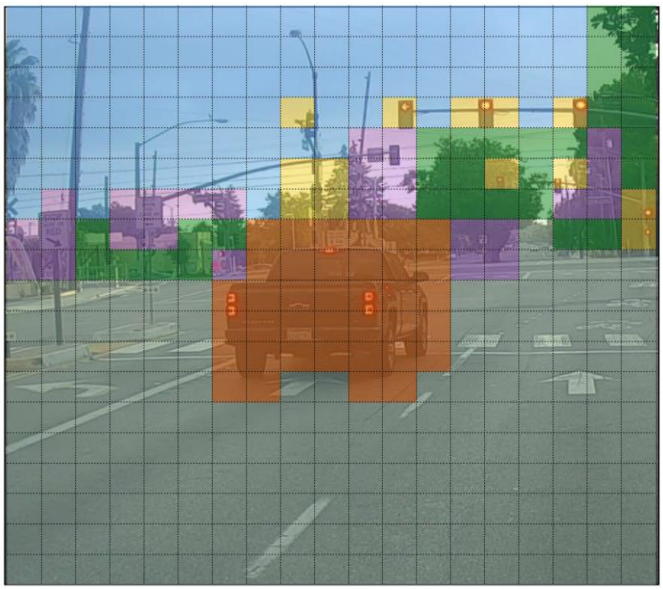
Gambar 2. 6 Sebelum dan Sesudah *Dropout*

E. YOLO (You Only Look Once)

YOLO (*You Only Look Once*) adalah algoritma yang menggunakan *convolutional neural network* (CNN) untuk mendeteksi objek secara *real time*. Algoritma YOLO mempunyai beberapa kelebihan dibandingkan dengan algoritma yang menggunakan *classifier*, Dilihat dari keseluruhan pada citra saat dilakukan percobaan dengan prediksi yang telah diinfokan pada citra. Sehingga hal tersebut membuat prediksi yang memuat jaringan saraf ini tidak seperti algoritma *Region Convolutional Neural Network* (R-CNN) yang ribuan citra dalam prosesnya sehingga membuat algoritma YOLO beberapa kali lebih cepat dibandingkan algoritma R-CNN. Algoritma YOLO bekerja menggunakan 3 teknik berikut:

1. Residual Blocks

Yaitu dengan membagi gambar atau *frame* menjadi beberapa bagian daerah, lalu daerah tersebut diberi *grid* dan setiap sel *grid* akan mendeteksi objek yang muncul di dalamnya. Pada Gambar 2. 7 misalnya jika objek muncul pada sel *grid* tertentu, maka sel tersebut akan bertanggung jawab untuk mendeteksinya.



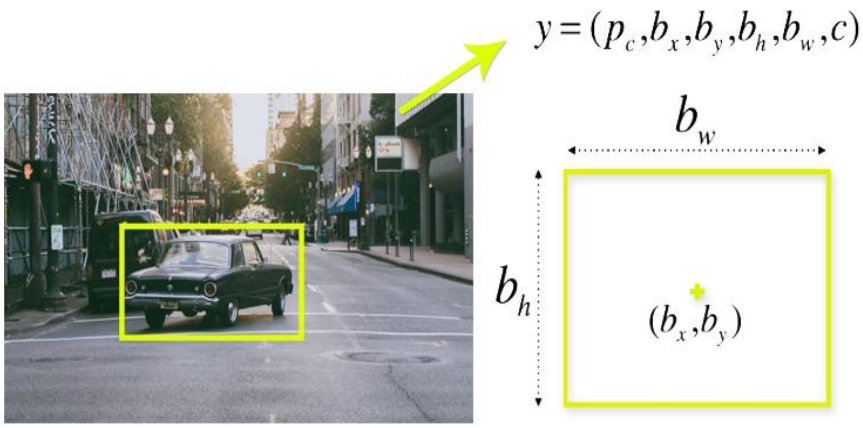
Gambar 2. 7 *Grid*

2. Bounding Box Regression

Yaitu kotak pembatas yang menyoroti objek dalam gambar, setiap kotak pembatas pada gambar terdiri dari atribut berikut:

a. Lebar (bw)
b. Tinggi (bh)
c. Kelas (misalnya mobil, motor, kulkas, dll.) ini diwakili oleh huruf c
d. Pusat kotak pembatas (bx, by)

Pada Gambar 2. 8 menunjukkan contoh kotak pembatas, Kotak pembatas telah diwakili oleh garis kuning.



Gambar 2. 8 *Bounding Box*

3. Intersection Over Union (IOU)

Yaitu metrik yang memungkinkan untuk mengevaluasi seberapa mirip kotak pembatas prediksi dengan kotak pembatas kebenaran dasar. Pada Gambar 2. 9 memberikan contoh sederhana tentang cara kerja IOU.



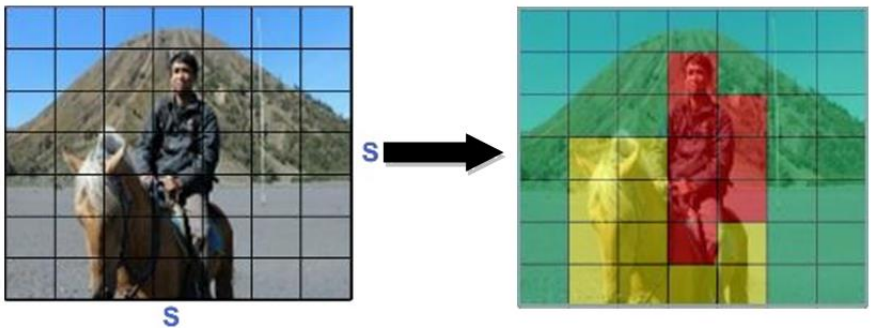
Gambar 2. 9 *Intersection Over Union (IOU)*

Pada Gambar 2. 9 ada dua kotak pembatas, satu berwarna hijau dan yang lainnya berwarna merah, Kotak merah adalah kotak pembatas prediksi sedangkan kotak hijau adalah kotak pembatas pembenaran dasar.

F. Rumus Algoritma YOLO

Terdapat tahapan dalam perumusan algoritma YOLO antara lain sebagai berikut:

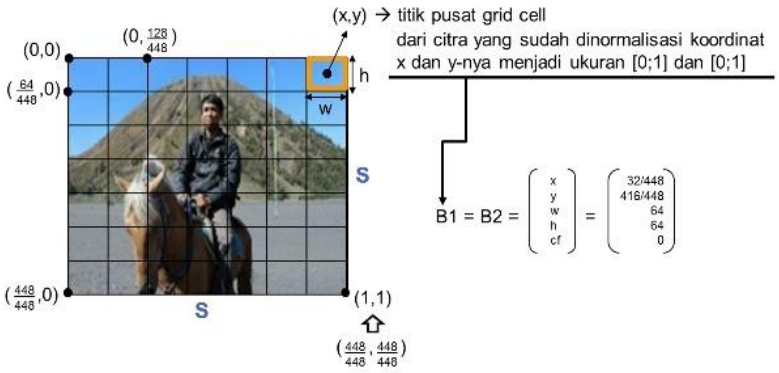
1. Membaca data citra dari berbagai ukuran.
2. Mengubah ukuran citra, Pada contoh Gambar 2.10 ukuran citra diubah menjadi 448x448 lalu membuat grid pada citra dengan ukuran S x S grid.



Gambar 2. 10 *Grid* Algoritma YOLO

Jika S=7, maka tiap *grid cell* ukurannya 64 x 64. Sehingga terdapat sebanyak 49 *grid cell.*

3. Tiap *grid cell* terdapat *Bounding Box* yang berisi 5 (lima) nilai, yaitu lokasi koordinat x berdasarkan baris (x), koordinat y berdasarkan kolom (y), lebar (w), tinggi (h), dan nilai *confidence* (x,y,w,h,cf) terhadap nilai *Bounding Box* yang ada, yaitu B1 dan B2.



Gambar 2. 11 *Grid* algoritma YOLO dengan nilai confidence

Confidence (cf) selalu di set = 0, jika dalam grid cell adalah background.

G. Darknet-53

DarkNet-53 adalah *Convolutional Neural Network* yang memiliki kedalaman 53 lapisan dan bertindak sebagai tulang punggung deteksi objek pada algoritma YOLO. *DarkNet-53* sering digunakan sebagai dasar untuk masalah deteksi objek dan alur kerja YOLO.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Gambar 2. 12 *Layer Darknet-53*

H. Python

Python adalah bahasa pemrograman yang memiliki multifungsi dan sangat sederhana sehingga mudah untuk digunakan dibandingkan bahasa pemrograman yang lainnya. Python dikenal sebagai bahasa yang menghubungkan kompetensi kemampuan susunan kode dengan sangat jelas dan memiliki pustaka standar yang besar dan berguna serta multifungsional. Python memiliki komunitas yang paling besar dibandingkan komunitas lainnya.

Python biasanya digunakan sebagai bahasa skrip meskipun pengguna dalam bahasa ini pada praktiknya mampu mencapai konteks pemanfaatan yang lebih luas dan biasanya tidak dilakukan dengan menggunakan bahasa skrip.

I. Google Colaboratory

Google Colaboratory atau biasa disebut dengan *Google Colab* adalah *tools* yang dikembangkan oleh Google untuk para pengembang yang menjalankan dan mengeksekusi code python di browser. *Google Colab* tidak seperti tools lainnya yang memerlukan konfigurasi karena menggunakan teknologi *cloud computing*, mampu mengakses mesin berkecepatan tinggi (GPU) dan sangat mudah dikoneksikan dengan perangkat lainnya seperti *Github* dan *Google drive*.

J. Flask

Flask adalah web *framework* yang sangat support dengan bahasa Python. Flask memiliki fungsi menampilan bahasa python dalam web. Melalui pemanfaatan flask, pengembang mampu membuat sebuah web dengan menggunakan bahasa python.

Flask termasuk dalam jenis *microframework* karena tidak seperti web *framework* lainnya yang menggunakan pustaka tertentu dalam pemanfatannya. Beberapa komponen yang biasa digunakan seperti *database*, form tampilan, dan lainnya tidak terpasang secara otomatis di flask. Sehingga komponen tersebut telah disediakan oleh pihak ketiga dan Flask mampu membuat suatu komponen tersebut seolah telah terpasang ke dalam flask itu sendiri.

K. API

API atau *Application Programming Interface* merupakan sekumpulan perintah dan protokol yang mempunyai fungsi untuk membangun dan merancang perangkat lunak pada suatu sistem operasi. Pengembang menggunakan fungsi standar yang memungkinkan API untuk berinteraksi dengan sistem operasi. API juga memiliki fitur yang terdiri dari antarmuka, struktur, kelas, dan fungsi yang dimanfaatkan untuk membuat suatu perangkat lunak. (Afrianto & Atin, 2017)

Dengan pemanfaatan API, memungkinkan pada pengembang untuk melakukan *reserve* suatu perangkat lunak yang telah dibuat yang kemudian mampu dikembangkan atau diimplementasikan ke dalam perangkat lunak yang lainnya. API juga memiliki fungsi menghubungkan suatu aplikasi dengan aplikasi lainnya. API memiliki keunggulan yang memungkinkan beberapa aplikasi untuk saling terhubung dan berinteraksi

L. CUDA

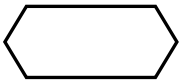
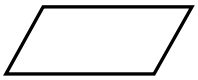
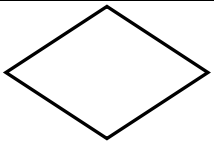
CUDA atau *Compute Unified Device Architecture* adalah platform komputasi paralel dan antarmuka pemrograman aplikasi yang dibuat dan dikembangkan oleh Nvidia. Hal ini untuk memungkinkan perangkat lunak menggunakan *graphics processing unit* (GPU) untuk tujuan umum. CUDA mampu melakukan komputasi secara bersamaan dan bersatu yaitu pada bagian CPU dan GPU. Hal ini menyebabkan kemampuan komputer menjadi luar biasa.

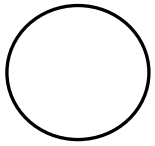
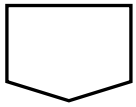
M. Flowchart

Flowchart atau yang biasa disebut bagan alur merupakan diagram yang memungkinkan untuk menampilkan suatu langkah dan keputusan sehingga mampu membuat proses dalam program.

Flowchart memiliki peran penting dalam memutuskan suatu langkah dari suatu proses uturan program yang memiliki keterlibatan dengan banyak pihak. Selain itu dengan pemanfaatan *flowchart* sehingga sebuah urutan dalam program akan singkat, padat, jelas dan mengurangi salah penafsiran.

Tabel 2. 1 Simbol *Flowchart*

Simbol	Nama	Fungsi
	*Terminator*	Awal / Akhir dari program
	*Garis alir* (flow line)	Alur proses program
	*Preparation*	Proses persiapan atau penetapan pada program
	Proses	Suatu proses dalam data program
	*Input* / *output* data	Berupa masukan, keluaran dan parameter pada data program
	*Predefined process* (sub program)	Awal / Akhir dari sub program
	*Decision*	Perbandingan atau penyaringan data program

	*On page connector*	Penghubung program dalam halaman yang sama
	*Off page connector*	Penghubung program dalam halaman yang berbeda

BAB III

METODE PENELITIAN

A. Jenis Penelitian

Penelitian mengadakan studi literatur dengan mempelajari buku-buku dan referensi dari website baik jurnal, skripsi, maupun situs-situs yang berhubungan dengan tugas akhir.

B. Lokasi Dan Waktu Penelitian

Lokasi penelitian di Kota Parepare, Kecamatan Ujung, Kelurahan Lapadde dan waktu penelitian akan dilaksanakan kurang lebih +4 bulan pada tahun 2021.

C. Teknik Pengumpulan Data

1. **Studi Pustaka**

   Mencari referensi yang berhubungan dengan topik yang akan dibuat. Pencarian referensi akan dicari di buku-buku, internet dan hasil penelitian maupun karya tulis ilmiah.

2. **Pengamatan (Observasi)**

   Pengamatan adalah metode pengumpulan data dengan cara mengadakan tinjauan secara langsung ke objek yang diteliti. Untuk mendapatkan data yang bersifat nyata dan meyakinkan maka penulis melakukan pengamatan.

D. Alat Dan Bahan Penelitian

Untuk melakukan proses penelitian dalam mendeteksi objek pada gambar, maka diperlukan perangkat keras dan perangkat lunak guna untuk mendukung kegiatan penelitian. Berikut ini merupakan perangkat keras dan perangkat lunak yang digunakan dalam mendeteksi objek pada gambar.

1. Perangkat lunak

Perangkat keras yang digunakan dalam melakukan proses penelitian deteksi objek adalah sebagai berikut :

Tabel 3. 1 Spesifikasi perangkat keras

Spesifikasi	
Merek laptop	Asus A456U
Processor laptop	Intel(R) Core(TM) i7-7500U, CPU @ 3.5 GHz
Ram laptop	8 (8192) GB

2. Perangkat lunak

Perangkat lunak yang digunakan dalam melakukan proses penelitian deteksi objek adalah sebagai berikut :

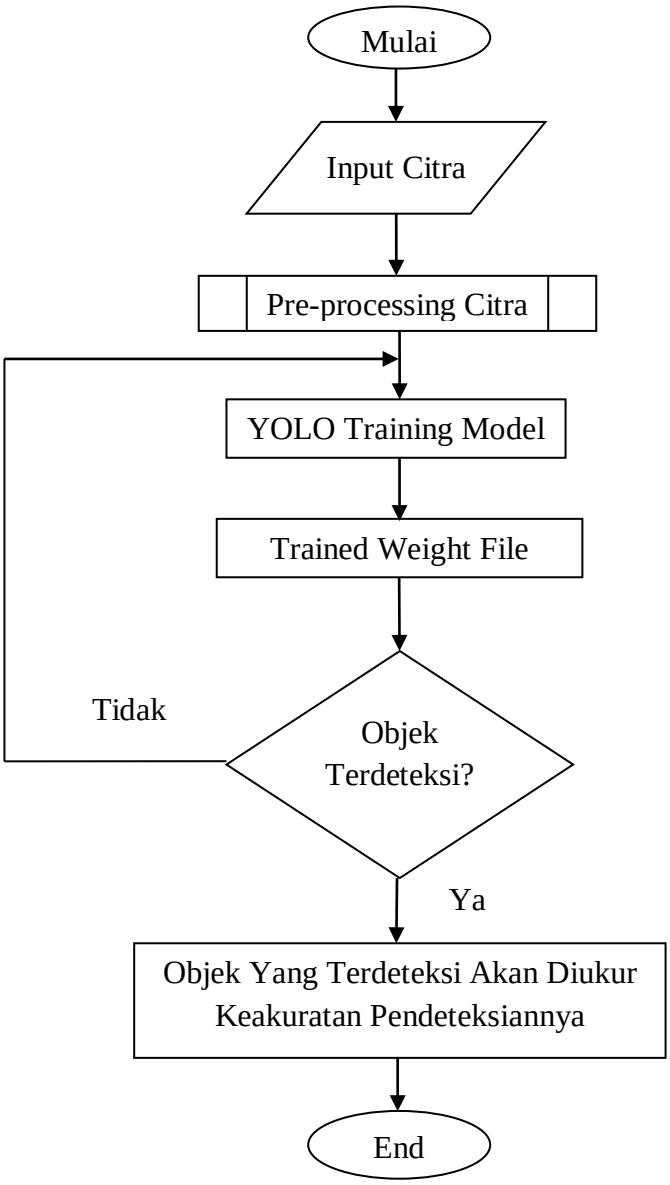
Tabel 3. 2 Spesifikasi perangkat lunak

Spesifikasi	
Sistem operasi	Windows 10
Tool pemrograman	Google Colab, Anaconda
Bahasa pemrograman	Python

E. Diagram Perancangan Proses

Diagram tahapan perancangan proses ditunjukan pada Gambar 3. 1. Ringkasan perancangan proses adalah sebagai berikut:

1. Input citra yang telah diperoleh secara manual menggunakan smartphone
2. Pra-processing citra yang terdiri dari kelas dan pelabelan citra.
3. YOLO training model merupakan proses konfigurasi dan pelatihan pada citra.
4. Trained weight file yaitu hasil output dari proses pelatihan yang berisi bobot berformat .weight.
5. Jika objek terdeteksi maka objek yang terdeteksi akan diukur keakuratan pendeteksiannya sedangkan jika objek tidak terdeteksi maka citra perlu dilakukan proses pelatihan ulang.



Gambar 3. 1 Diagram alur Perancangan Proses

BAB IV

HASIL DAN PEMBAHASAN

A. Perancangan Data

Proses pendeteksian objek pada gambar menggunakan algoritma YOLO memerlukan sejumlah besar data citra untuk dilatih, sehingga mendapatkan bobot optimal dalam pendeteksian dan pengklasifikasian objek. Selain itu diperlukan pula sejumlah data citra untuk menguji keakuratan pendeteksian objek.

1. Data Awal

Sebelum melakukan proses pelatihan dan proses pengujian, langkah pertama yang perlu dilakukan adalah mempersiapkan data citra yang dibutuhkan pada penelitian ini. Berdasarkan batasan masalah, pengumpulan data citra diperoleh secara manual menggunakan *smartphone*, sehingga keterangan objek untuk setiap citra perlu dibangun terlebih dahulu. Keterangan yang dimaksud merupakan informasi tentang letak objek dan kelas objek yang akan dideteksi dan diklasifikasi. Keterangan ini berfungsi sebagai target atau acuan untuk memperoleh bobot pada saat proses pelatihan data dan menjadi pembanding nilai output pada saat proses pengujian.

2. *Pre-trained Weights*

Dalam proses pelatihan menggunakan model dari *darknet open source*, diperlukan bobot awal dari sumber yang sama. Dalam pelatihan ini bobot awal yang digunakan bukan data random atau acak, tetapi data yang dilatih

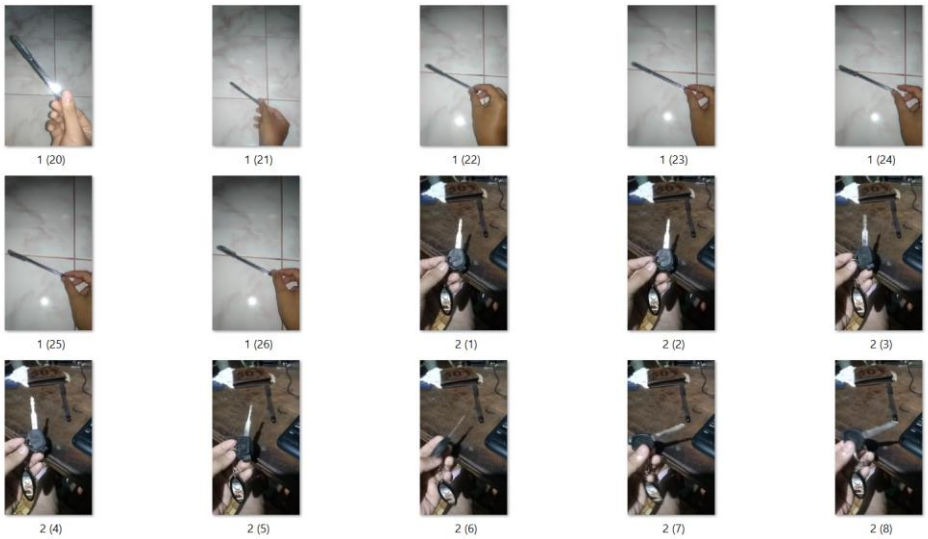
sebelumnya pada jaringan YOLO untuk data citra. Data *pre-trained Weights* pada penelitian ini menggunakan model *yolov4-tiny* yang di unduh dari https://github.com/AlexeyAB/darknet/ dengan format file .weights.

B. Perancangan Proses

Secara umum proses deteksi objek pada gambar menggunakan algoritma YOLO dapat dilihat pada diagram alur dalam Gambar 3. 1. Rincian tentang perancangan proses deteksi objek pada gambar menggunakan algoritma YOLO ini yaitu sebagai berikut.

1. Input citra

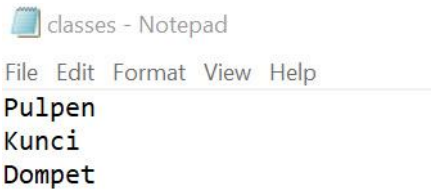
Data citra yang diinput merupakan citra yang diperoleh secara manual menggunakan *smartphone*. Jumlah data citra yang perlu dikumpulkan untuk proses pelatihan dan pengujian minimal 1 (satu) citra.



Gambar 4. 1 Citra yang diinput dari *smartphone*

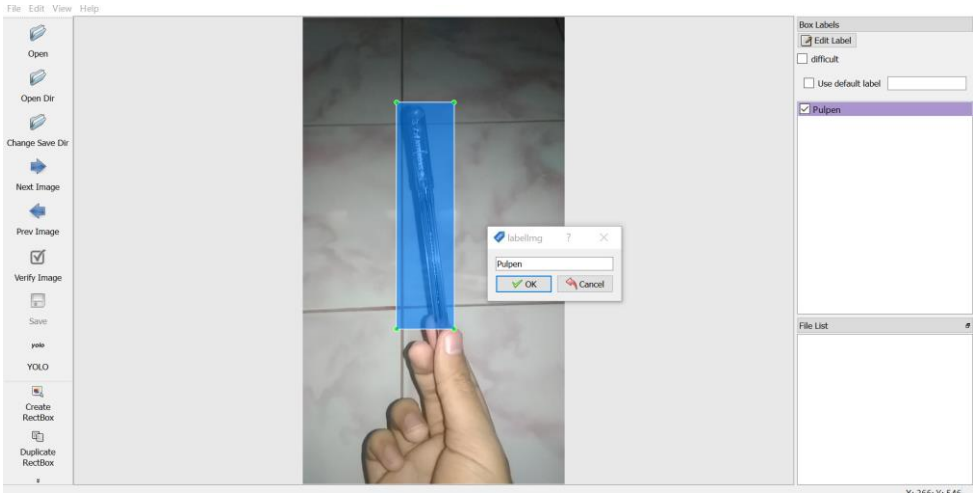
2. *Pre-processing* Citra

Pada tahap ini dilakukan *pre-processing* citra yang terdiri dari kelas dan pelabelan pada citra, Kelas berjumlah sesuai dengan jenis citra yang akan dideteksi. Setelah menentukan kelas, citra yang sudah diinput akan masuk ke proses pelabelan citra.



Gambar 4. 2 Kelas untuk pelabelan

Pelabelan citra adalah tahap awal yang dimana setiap data citra yang akan diberikan label mempunyai tujuan untuk menyimpan informasi pada citra. Proses pelabelan citra yaitu dengan cara memberikan *bounding box* serta *classes* pada objek citra yang tersorot. Pelabelan citra menggunakan *LabelImg* yang merupakan salah satu tools atau modul *Python*.



Gambar 4. 3 Pelabelan citra

Hasil dari pelabelan citra akan menghasilkan *output file* yang berisikan urutan objek dan koordinat (x, y, width, weight) pada citra.



Gambar 4. 4 Hasil pelabelan citra

3. YOLO Training Model

Setelah dilakukan *pre-processing* citra, Selanjutnya akan dilakukan proses konfigurasi jaringan YOLO. Unduh Darknet terlebih dahulu dan aktifkan GPU dan depency lainnya pada perangkat, seperti OpenCV dan *library* lainnya untuk melakukan *build* pada Darknet yang akan digunakan untuk melakukan proses *training*.

```
[ ]  # unduh darknet
     !git clone https://github.com/AlexeyAB/darknet
```

Gambar 4. 5 Unduh Darknet

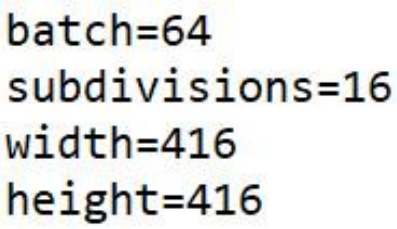
```
[ ]  # mengubah makefile untuk mengaktifkan GPU,CUDNN,CUDNN_HALF,LIBSO dan OPENCV
     %cd /content/darknet/
     !sed -i 's/OPENCV=0/OPENCV=1/' Makefile
     !sed -i 's/GPU=0/GPU=1/' Makefile
     !sed -i 's/CUDNN=0/CUDNN=1/' Makefile
     !sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
     !sed -i 's/LIBSO=0/LIBSO=1/' Makefile
```

Gambar 4. 6 Mengaktifkan GPU,cuDNN dan openCV

```
[ ]  # build darknet
     !make
```

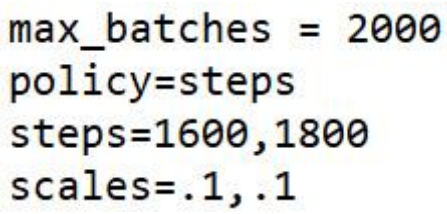
Gambar 4. 7 *Build* Darknet

Setelah melakukan *build* pada Darknet, selanjutnya terlebih dahulu melakukan konfigurasi dan pembuatan model pada modul. Untuk melakukan proses *training* pada data, melakukan pengaturan pada *batch* sebanyak 64 *batch* dan *subdivisions* sesuai pada perangkat yang cocok dan mengubah semua ukuran pada citra menjadi 416x416 sesuai dengan ketentuan pada YOLO. (Redmon et al., 2016).



Gambar 4. 8 Mengatur *batch*, *subdivisions*, *width*, dan *height*

Banyaknya *max batches* yang diperlukan harus ditentukan. Jumlah *max batches* dalam ketentuan YOLO adalah 2000*jumlah kelas yang akan di proses. Kemudian melakukan penentuan dalam akurasi pada pelatihan, yaitu 80%—90% dari total *max batches* sesuai ketentuan YOLO. (Redmon et al., 2016).



Gambar 4. 9 Mengatur *max batches* dan *steps*

Selanjutnya melakukan konfigurasi jumlah pada kelas (*classes*) dan *filtering* ((Jumlah Kelas + width + height + x + y + confidence) * num)) pada YOLO

yang dapat dilihat pada Gambar 4. 10. Cara menentukan jumlah kelas (*classes*) dan *filters* adalah sebagai berikut:

filters = (jumlah kelas + 5) * 3

= (jumlah kelas + width + height + x + y + confidence) * num

= (jumlah kelas + 1 + 1 + 1 + 1 + 1) * num

= (jumlah kelas + 5) * num

num = jumlah mask dalam layer YOLO

= (jumlah kelas + 5) * 3. (Redmon et al., 2016).

```
[convolutional]
size=1
stride=1
pad=1
filters=18
activation=linear

[yolo]
mask = 1,2,3
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319
classes=1
```

Gambar 4. 10 Mendefinisikan jumlah kelas dan *filtering*

Setelah melakukan konfigurasi pada *filtering* dan jumlah kelas, selanjutnya masuk ke dalam tahap memasukkan *backup path* dan jumlah kelas yang akan digunakan untuk proses *training*. Gambar 4. 11 adalah jumlah kelas yang akan dilakukan proses pelatihan dan file *backup* yang digunakan untuk menyimpan hasil output dari proses *training*.

```
classes = 1
train  = data/train.txt
valid  = data/test.txt
names = data/obj.names
backup = /mydrive/yolov4-tiny/training
```

Gambar 4. 11 Mendefinisikan *backup path* dan jumlah kelas

Gambar 4. 12 adalah file konfigurasi model terakhir yang diperlukan sebelum memulai proses *training*. File konfigurasi model ini berfungsi untuk menahan *path* relatif pada semua citra dalam proses *training*. Hal ini dilakukan agar iterasi pada setiap citra dengan ekstensi jpg memproses setiap citra menjadi data *train*. Yang kemudian melakukan proses *generate* pada file untuk memuat setiap *path* pada citra yang diiterasikan.

```
import glob, os

# Current directory
current_dir = os.path.dirname(os.path.abspath(__file__))

print(current_dir)

current_dir = 'data/obj'

# Percentage of images to be used for the test set
percentage_test = 10;

# Create and/or truncate train.txt and test.txt
file_train = open('data/train.txt', 'w')
file_test = open('data/test.txt', 'w')

# Populate train.txt and test.txt
counter = 1
index_test = round(100 / percentage_test)
for pathAndFilename in glob.iglob(os.path.join(current_dir, "*.jpg")):
    title, ext = os.path.splitext(os.path.basename(pathAndFilename))

    if counter == index_test:
        counter = 1
        file_test.write("data/obj" + "/" + title + '.jpg' + "\n")
    else:
        file_train.write("data/obj" + "/" + title + '.jpg' + "\n")
        counter = counter + 1
```

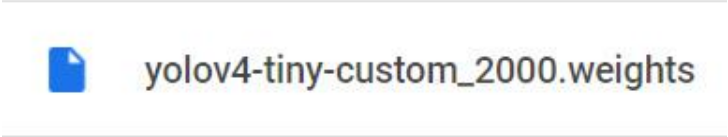
Gambar 4. 12 Generate_train.py

Setelah semua konfigurasi model telah dikonfigurasi dan dibuat, Selanjutnya melakukan proses *training* data citra pada model yang telah disiapkan.

```
# menjalankan proses kostum training detektor
!./darknet detector train data/obj.data cfg/yolov4-tiny-custom.cfg yolov4-tiny.conv.29 -dont_show -map
```

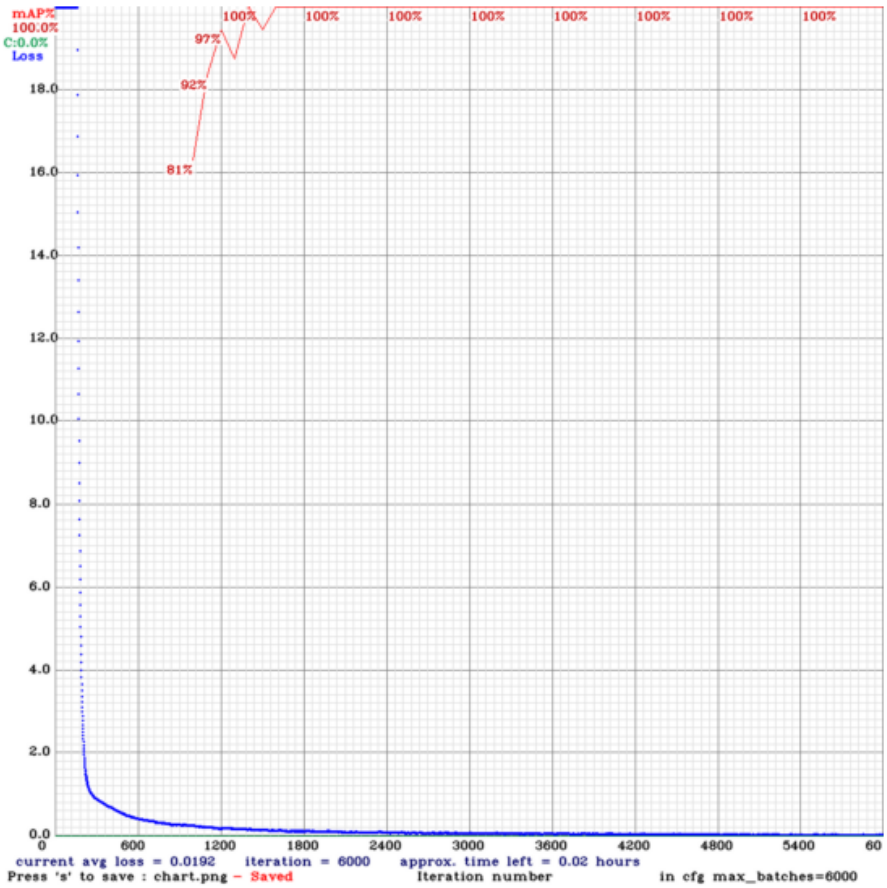
Gambar 4. 13 Perintah untuk melakukan *training*

Hasil dari *training* akan menghasilkan *trained weight* yang masuk pada folder *backup* seperti pada Gambar 4. 14.



Gambar 4. 14 Hasil *training*

Setelah menghasilkan *trained weight*, Selanjutnya memilih batch terbaik yang memiliki keakuratan mAP (mean Average Precision) tertinggi dengan melihat proses *training* data melalui *chart* seperti pada Gambar 4. 15.



Gambar 4. 15 *Chart* proses *training*

C. Pengujian

1. Dataset Pengujian

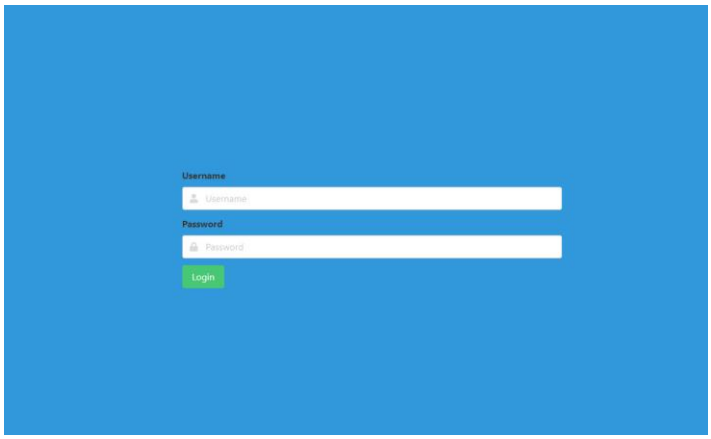
Sebelum melewati proses pengujian, Dataset pengujian perlu disiapkan terlebih dahulu. Sebanyak 10 citra yang digunakan untuk proses pengujian, banyaknya citra yang digunakan untuk mengetahui sejauh mana keakuratan model dalam mendeteksi suatu objek. Tabel 4. 1 menunjukkan rincian dari dataset yang digunakan saat proses pengujian.

Tabel 4. 1 Dataset Proses Pengujian

No	Jenis Objek	Total Citra Yang Diuji
1.	Kartu Smartfren	10
2.	Kartu Indosat	10
3.	Kartu Tri	10
4.	Benang Biru	10
5.	Benang Ungu	10
6.	Benang Pink	10
7.	Mobil Hijau	10
8.	Mobil Biru	10
9.	Mobil Putih	10
10.	Pulpen Snowman V7	10
11.	Pulpen Esco	10
12.	Pulpen Yaomei	10
13.	Rexona Orange	10
14.	Rexona Pink	10
15.	Rexona Abu-abu	10

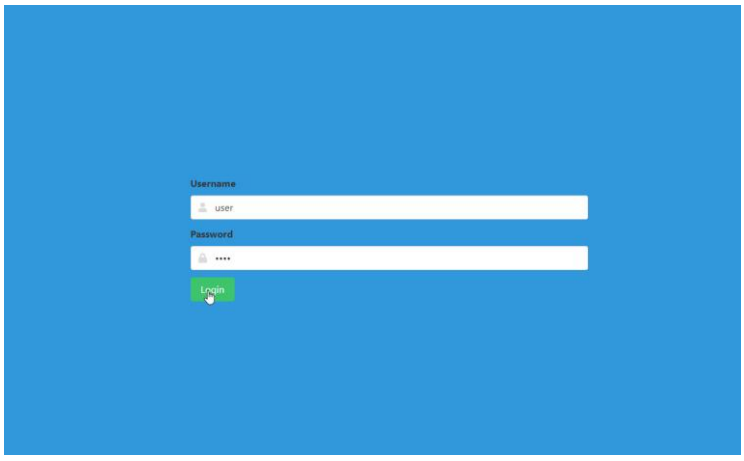
2. Implementasi Aplikasi Pengujian

Proses pengujian diimplementasikan pada perangkat lunak untuk mengetahui hasil kinerja metode. Pada subbab ini akan disajikan penjelasan aplikasi pengujian.



Gambar 4. 16 Tampilan Login

Pada Gambar 4. 16 merupakan tampilan Login dari aplikasi pengujian, Langkah pertama yang dilakukan adalah dengan memasukkan username dan password yang sudah disediakan.



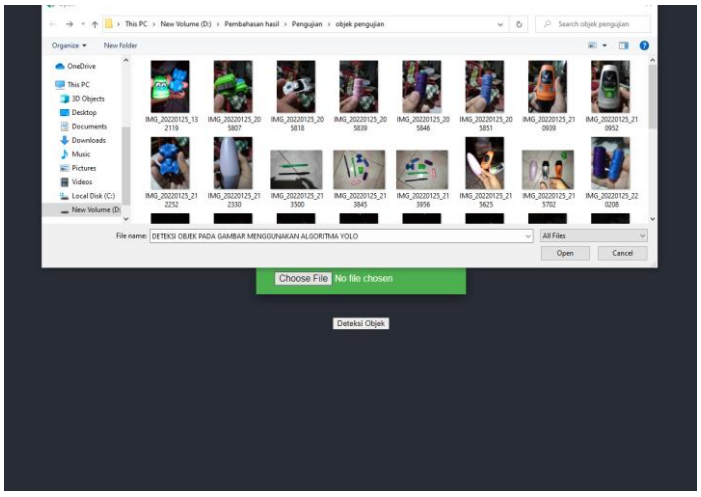
Gambar 4. 17 Memasukkan Username dan Password

Setelah memasukkan username dan password seperti pada Gambar 4. 17, selanjutnya masuk ke halaman deteksi objek dengan menekan tombol login.



Gambar 4. 18 Tampilan Halaman Deteksi Objek

Pada Gambar 4. 18. Merupakan tampilan halaman deteksi objek dari aplikasi pengujian, Langkah pertama dalam melakukan proses deteksi objek adalah menginput dataset citra yang akan diuji dengan menekan tombol *choose File*.



Gambar 4. 19 Menginput dataset citra yang akan diuji

Setelah melakukan penginputan dataset citra yang akan diuji, selanjutnya akan menampilkan dataset citra yang telah diinput seperti pada Gambar 4.20.



Gambar 4. 20 Dataset citra yang telah diinput

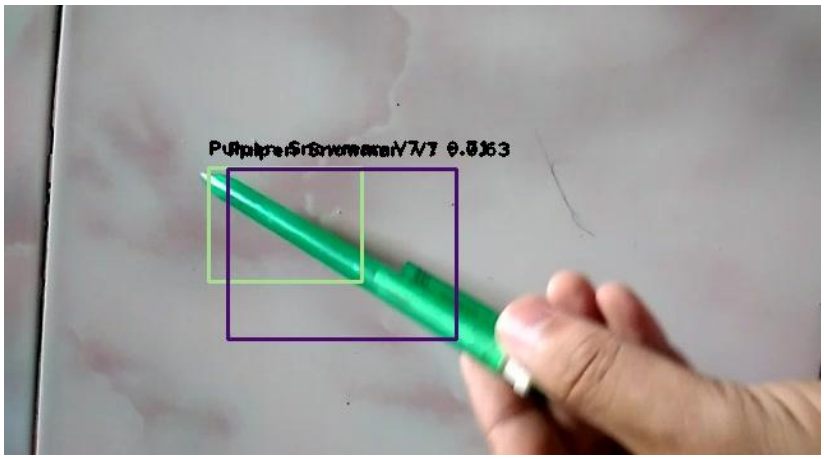
Langkah terakhir yang dilakukan adalah dengan menekan tombol deteksi objek yang secara otomatis akan mendeteksi objek pada gambar yang telah diinput seperti pada Gambar 4. 21.



Gambar 4. 21 Dataset citra yang terdeteksi

3. *Threshold*

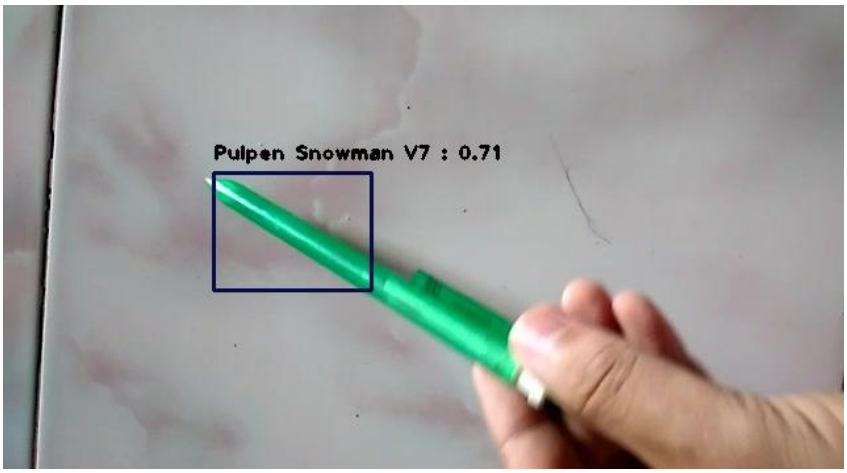
Beberapa dataset yang dijelaskan sebelumnya digunakan untuk proses pengujian. Hasil dalam proses pengujian yaitu dengan munculnya prediksi kotak pada pembatas, kelas, serta nilai *threshold* pada objek dalam citra. Sehingga dalam proses pengujian perlu diperhatikan suatu nilai *threshold* sebagai batas nilai kepercayaan kotak pembatas yang memuat suatu objek pada kotak pembatas. Jika nilai *threshold* pengujian terlalu rendah atau bahkan tidak terdeteksi, bisa jadi satu objek yang dideteksi memiliki beberapa kotak pembatas. Contoh kasus tersebut bisa dilihat pada Gambar 4. 20 dengan nilai *threshold* 0.7.



Gambar 4. 22 Kotak pembatas yang ganda

Pada Gambar 4. 20, hasil pengujian menunjukan bahwa terdapat 2 objek yang terdeteksi. Namun pada kenyataanya hanya terdapat 1 objek citra. Hal tersebut bisa terjadi dikarenakan nilai keyakinan pada bebrapa kotak pembatas lebih tinggi dari nilai *threshold* yang ditentukan. Sehingga kotak

pembatas mendapatkan hasil tersebut. Contoh citra dengan pendeteksian yang tepat dan sesuai dapat dilihat pada Gambar 4. 21 dengan nilai *threshold* 0.5.

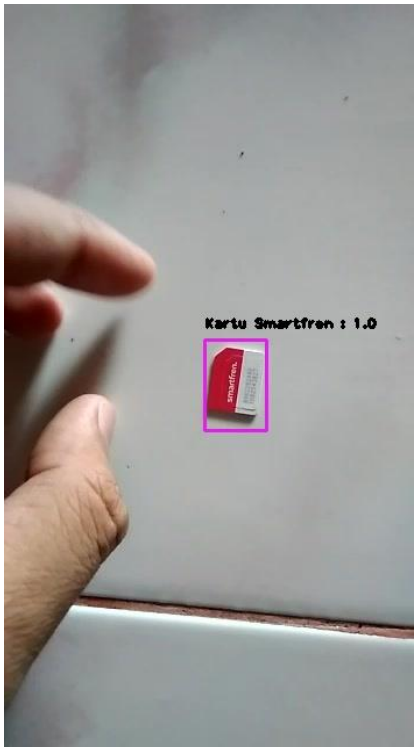


Gambar 4. 23 Kotak pembatas yang sesuai

Berdasarkan 2 contoh diatas, perlunya penyesuaian nilai threshold dalam suatu dataset pada citra agar objek bisa dikenali. Untuk mengatasi masalah pada penentuan nilai threshold, penelitian ini menggunakan nilai *threshold* yang rendah dan akan mengambil nilai kepercayaan kotak pembatas yang paling tinggi sebagai hasil dari deteksi.

4. *Output* Pengujian

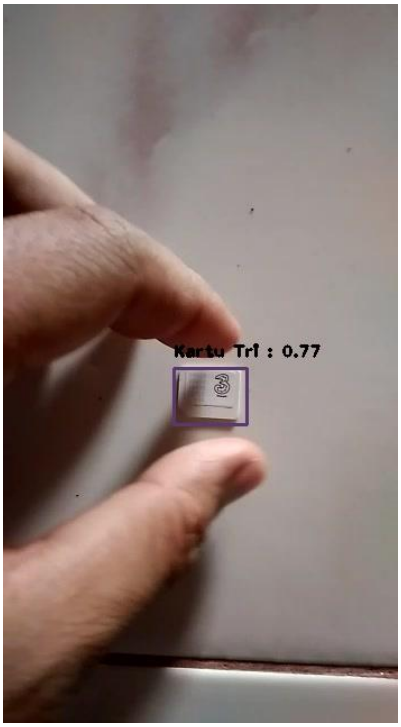
Setelah mengatur nilai *threshold*, proses pengujian dilakukan dengan menguji seluruh dataset pengujian. Beberapa gambar dibawah merupakan pasangan masukan dan keluaran data yang dapat dikenali. Dapat dilihat bahwa seluruh objek pada gambar dibawah ini berhasil dikenali dengan tepat dan memperoleh deteksi kotak pembatas yang sesuai.



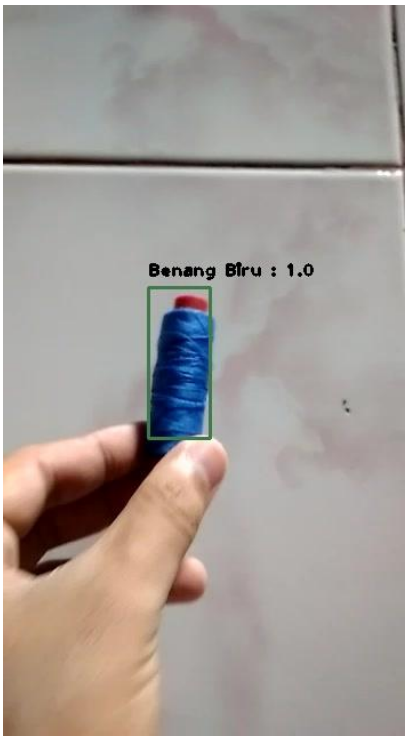
Gambar 4. 24 Hasil pengujian citra pada Kartu Smartfren



Gambar 4. 25 Hasil pengujian citra pada Kartu Indosat



Gambar 4. 26 Hasil pengujian citra pada Kartu Tri



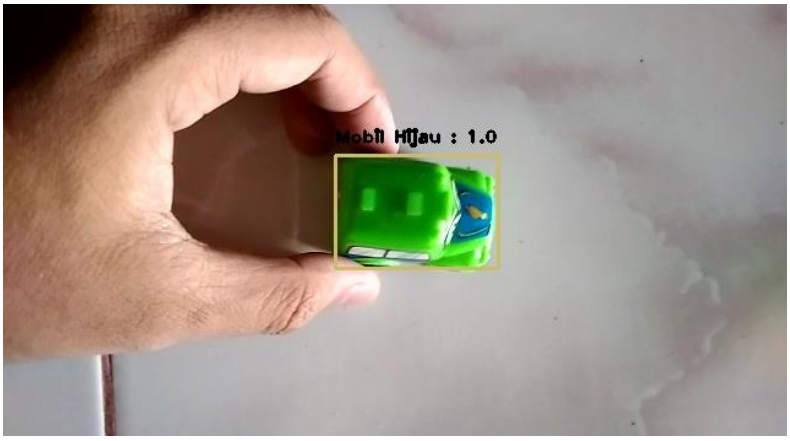
Gambar 4. 27 Hasil pengujian citra pada Benang Biru



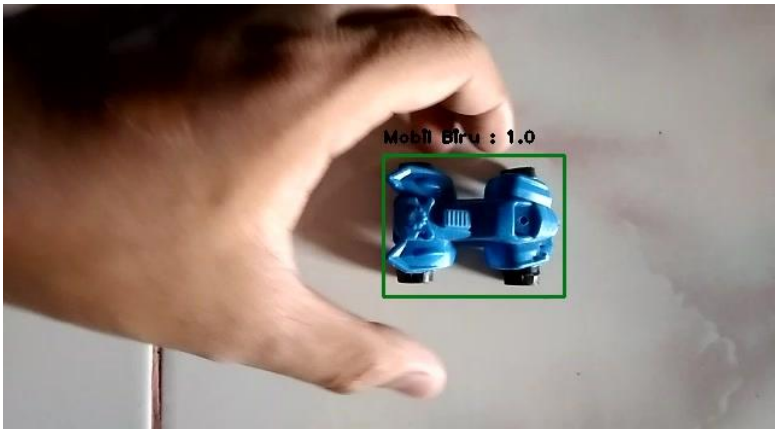
Gambar 4. 28 Hasil pengujian citra pada Benang Ungu



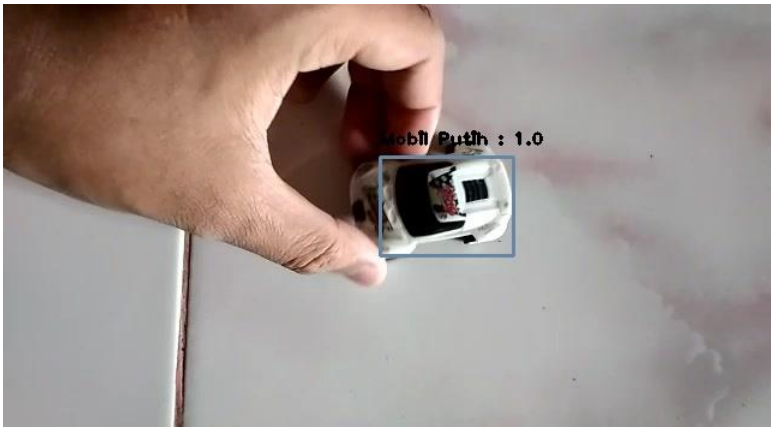
Gambar 4. 29 Hasil pengujian citra pada Benang Pink



Gambar 4. 30 Hasil pengujian citra pada Mobil Hijau



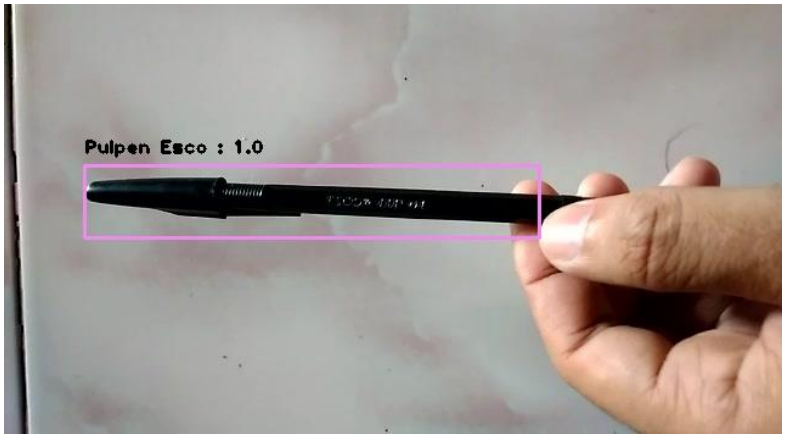
Gambar 4. 31 Hasil pengujian citra pada Mobil Biru



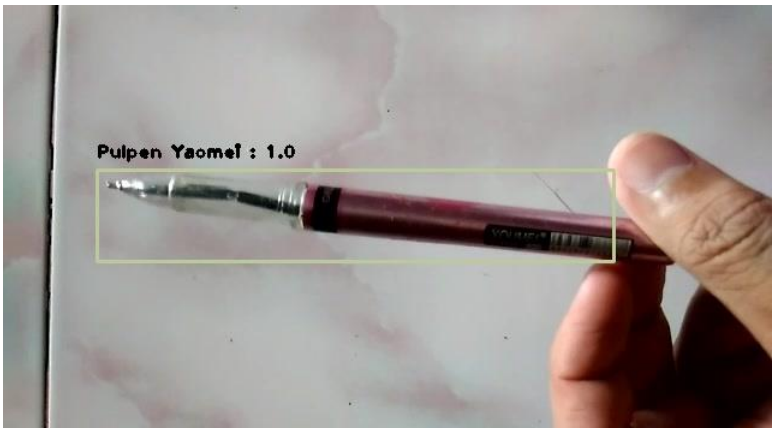
Gambar 4. 32 Hasil pengujian citra pada Mobil Putih



Gambar 4. 33 Hasil pengujian citra pada Pulpen Snowman V7



Gambar 4. 34 Hasil pengujian citra pada Pulpen Esco



Gambar 4. 35 Hasil pengujian citra pada Pulpen Yaomei



Gambar 4. 36 Hasil pengujian citra pada Rexona Orange



Gambar 4. 37 Hasil pengujian citra pada Rexona Pink



Gambar 4. 38 Hasil pengujian citra pada Rexona Abu-abu

5. Kinerja Metode YOLO

Kinerja Metode YOLO pada penelitian ini dipengaruhi oleh nilai threshold, Nilai tersebut diberikan kepada seluruh dataset pengujian. Kinerja metode ditentukan oleh klasifikasi objek. Pada Tabel 4. 2 sampai Tabel 4. 16 merupakan rincian hasil uji coba dataset.

Tabel 4. 2 Hasil uji coba dataset pengujian pada Kartu Smartfren

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.1	46.77%
2	0.5	0.09	48.12%
3	0.5	0.07	86.01%
4	0.5	0.2	0%
5	0.5	0.1	34.58%
6	0.5	0.1	65.05%
7	0.5	0.06	33.49%
8	0.5	0.3	92.23%
9	0.5	0.1	0%
10	0.5	0.1	26.31%
Nilai Rata-rata		0.122	43.37%
Standar Deviasi		0.004776	-
Standar Error		0.0004776	-

Tabel 4. 3 Hasil uji coba dataset pengujian pada Kartu Indosat

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.07	55.08%
2	0.5	0.06	74.21%
3	0.5	0.08	0%
4	0.5	0.09	0%
5	0.5	0.2	44.99%
6	0.5	0.1	14.45%
7	0.5	0.06	45.69%
8	0.5	0.2	57.77%
9	0.5	0.1	66.07%
10	0.5	0.1	0%
Nilai Rata-rata		0.106	35.79%
Standar Deviasi		0.002424	-
Standar Error		0.0002424	-

Tabel 4. 4 Hasil uji coba dataset pengujian pada Kartu Tri

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.06	92.57%
2	0.5	0.08	84.33%
3	0.5	0.06	94.32%
4	0.5	0.06	77.85%
5	0.5	0.05	99.44%
6	0.5	0.06	82.13%
7	0.5	0.09	54.99%
8	0.5	0.2	0%
9	0.5	0.09	67.01%
10	0.5	0.1	0%
Nilai Rata-rata		0.085	65,26%
Standar Deviasi		0.001725	-
Standar Error		0.0001725	-

Tabel 4. 5 Hasil uji coba dataset pengujian pada Benang Biru

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0,08	94.89%
2	0.5	0.1	98.66%
3	0.5	0.1	76.51%
4	0.5	0.04	99.99%
5	0.5	0.09	84.68%
6	0.5	0.08	73.23%
7	0.5	0.1	54.67%
8	0.5	0.1	61.11%
9	0.5	0.1	46.88%
10	0.5	0.1	53.95%
Nilai Rata-rata		0.079	74.45%
Standar Deviasi		0.000429	-
Standar Error		0.0000429	-

Tabel 4. 6 Hasil uji coba dataset pengujian pada Benang Ungu

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.1	78.85%
2	0.5	0.08	66.78%
3	0.5	0.08	77.77%
4	0.5	0.1	67.12%
5	0.5	0.1	88.98%
6	0.5	0.09	80.02%
7	0.5	0.2	34.78%
8	0.5	0.1	0%
9	0.5	0.07	57.46%
10	0.5	0.1	43.39%
Nilai Rata-rata		0.102	59.51%
Standar Deviasi		0.0011276	-
Standar Error		0.00011276	-

Tabel 4. 7 Hasil uji coba dataset pengujian pada Benang Pink

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.1	95.55%
2	0.5	0.2	68.11%
3	0.5	0.2	56.17%
4	0.5	0.1	79.90%
5	0.5	0.1	93.38%
6	0.5	0.1	57.84%
7	0.5	0.1	36.29%
8	0.5	0.09	0%
9	0.5	0.07	41.07%
10	0.5	0.1	68.92%
Nilai Rata-rata		0.186	59.72%
Standar Deviasi		0.006744	-
Standar Error		0.0006744	-

Tabel 4. 8 Hasil uji coba dataset pengujian pada Mobil Hijau

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.07	86.43%
2	0.5	0.07	44.56%
3	0.5	0.1	57.35%
4	0.5	0.2	67.41%
5	0.5	0.1	67,88%
6	0.5	0.1	94.80%
7	0.5	0.1	37.17%
8	0.5	0.1	71.92%
9	0.5	0.1	23.88%
10	0.5	0.08	39.64%
Nilai Rata-rata		0.102	59.10%
Standar Deviasi		0.001216	-
Standar Error		0.0001216	-

Tabel 4. 9 Hasil uji coba dataset pengujian pada Mobil Biru

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.1	62.89%
2	0.5	0.1	71.90%
3	0.5	0.1	98.44%
4	0.5	0.1	65.71%
5	0.5	0.1	48.84%
6	0.5	0.1	35.95%
7	0.5	0.1	0%
8	0.5	0.1	87.29%
9	0.5	0.1	74.88%
10	0.5	0.09	99.32%
Nilai Rata-rata		0.099	64.52%
Standar Deviasi		0.000009	-
Standar Error		0.0000009	-

Tabel 4. 10 Hasil uji coba dataset pengujian pada Mobil Putih

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.09	96.66%
2	0.5	0.09	76.91%
3	0.5	0.09	78.07%
4	0.5	0.08	23.77%
5	0.5	0.04	41.94%
6	0.5	0.1	92.33%
7	0.5	0.3	91.02%
8	0.5	0.3	89.01%
9	0.5	0.05	41.05%
10	0.5	0.1	47.83%
Nilai Rata-rata		0.124	67.85%
Standar Deviasi		0.008104	-
Standar Error		0.0008104	-

Tabel 4. 11 Hasil uji coba dataset pengujian pada Pulpen Snowman V7

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.2	0%
2	0.5	0.1	98.77%
3	0.5	0.1	98.56%
4	0.5	0.1	0%
5	0.5	0.1	78.76%
6	0.5	0.09	71.01%
7	0.5	0.1	83.12%
8	0.5	0.2	94.17%
9	0.5	0.1	0%
10	0.5	0.06	0%
Nilai Rata-rata		0.115	52.43%
Standar Deviasi		0.001945	-
Standar Error		0.0001945	-

Tabel 4. 12 Hasil uji coba dataset pengujian pada Pulpen Esco

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.1	0%
2	0.5	0.1	99.34%
3	0.5	0.1	99.66%
4	0.5	0.2	0%
5	0.5	0.1	86.90%
6	0.5	0.1	0%
7	0.5	0.1	71.91%
8	0.5	0.2	54.34%
9	0.5	0.1	0%
10	0.5	0.1	97.18%
Nilai Rata-rata		0.120	50.93%
Standar Deviasi		0.0016	-
Standar Error		0.00016	-

Tabel 4. 13 Hasil uji coba dataset pengujian pada Pulpen Yaomei

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.2	84.76%
2	0.5	0.07	51.33%
3	0.5	0.2	49.60%
4	0.5	0.2	51.94%
5	0.5	0.3	95.91%
6	0.5	0.05	0%
7	0.5	0.04	37.25%
8	0.5	0.2	0%
9	0.5	0.1	0%
10	0.5	0.3	55.59%
Nilai Rata-rata		0.166	42.63%
Standar Deviasi		0.008344	-
Standar Error		0.0008344	-

Tabel 4. 14 Hasil uji coba dataset pengujian pada Rexona Orange

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.03	38.66%
2	0.5	0.05	75.44%
3	0.5	0.05	78.93%
4	0.5	0.09	81.06%
5	0.5	0.05	79.60%
6	0.5	0.08	43.91%
7	0.5	0.1	43.02%
8	0.5	0.2	0%
9	0.5	0.07	22.62%
10	0.5	0.08	0%
Nilai Rata-rata		0.08	46.32%
Standar Deviasi		0.00202	-
Standar Error		0.000202	-

Tabel 4. 15 Hasil uji coba dataset pengujian pada Rexona Pink

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.08	0%
2	0.5	0.1	0%
3	0.5	0.1	92.33%
4	0.5	0.2	92.49%
5	0.5	0.1	98.32%
6	0.5	0.1	91.80%
7	0.5	0.08	99.01%
8	0.5	0.07	68.82%
9	0.5	0.09	52.83%
10	0.5	0.3	31.56%
Nilai Rata-rata		0.122	62.71%
Standar Deviasi		0.004696	-
Standar Error		0.0004696	-

Tabel 4. 16 Hasil uji coba dataset pengujian pada Rexona Abu-abu

No Pengujian	Threshold	Waktu Deteksi	Confidence
1	0.5	0.1	88.78%
2	0.5	0.1	68.11%
3	0.5	0.1	93.45%
4	0.5	0.3	78.02%
5	0.5	0.05	94.51%
6	0.5	0.07	97.92%
7	0.5	0.05	71.24%
8	0.5	0.05	0%
9	0.5	0.08	68.14%
10	0.5	0.1	51.60%
Nilai Rata-rata		0.1	711.77%
Standar Deviasi		0.00488	-
Standar Error		0.000488	-

BAB IV

PENUTUP

A. Kesimpulan

Pada penelitian ini, penulis telah berhasil membangun aplikasi pengujian untuk mendeteksi objek menggunakan algoritma YOLO. Penyelesaian proses ini menggunakan model dan *trained weight*. Berdasarkan pengujian yang dilakukan pada citra telah berhasil melakukan pendeteksian pada objek. Sehingga memungkinkan untuk dimanfaatkan pada pengembangan dibidang tertentu seperti di bidang keamanan dengan menerapkannya ke dalam kamera CCTV.

B. Saran

Pada rancangan deteksi objek pada gambar menggunakan algoritma YOLO yang telah dilakukan, masih terdapat beberapa kekurangan yang penulis harapkan dapat disempurnakan pada penelitian berikutnya, seperti :

1. Dataset pelatihan yang digunakan pada penelitian ini cukup sedikit dan memiliki beberapa gangguan. Sehingga penambahan data yang banyak dan penangkapan citra yang baik dan tanpa gangguan akan mendapatkan hasil yang lebih akurat sehingga objek dapat terdeteksi dengan baik.
2. Terkadang suatu objek gagal terdeteksi, meleset ataupun salah deteksi objek. Ini disebabkan karena penangkapan citra baik untuk proses pelatihan maupun pengujian memiliki beberapa gangguan.

DAFTAR PUSTAKA

Afrianto, I., & Atin, S. (2017). Rancang bangun model agregator jurnal online. Seminar Nasional Aplikasi Teknologi Informasi. *SNATi*. D: 9-16.

Anka, A. (2020). *YOLO v4: Optimal Speed & Accuracy for object detection*. Diperoleh Dari:https://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection-79896ed47b50. (Diakses 10 Juli 2021).

Chablani, M. (2017). *YOLO — You only look once, real time object detection explained*. Diperoleh dari:https://towardsdatascience.com/yolo-you-only-look-once-real-time-object-detection-explained-492dc9230006. (Diakses 8 April 2021).

Gautama, T. K., Hendrik, A., & Riskadewi. (2016). Pengenalan Objek pada Computer Vision dengan Pencocokan Fitur Menggunakan Algoritma SIFT Studi Kasus: Deteksi Penyakit Kulit Sederhana. *JuTISI*. 2(3): 437-459.

Hidayatulloh, Muhammad Syarif. (2021). *Sistem Pengenalan Wajah Menggunakan Metode Yolo (You Only Look Once)*. Skripsi tidak dipublikasikan. Surabaya: Universitas Dinamika.

James, A. (2021). *Faster YOLOv4 Performance with CUDA enabled OpenCV*. Diperoleh dari:https://towardsdatascience.com/yolov4-with-cuda-powered-opencv-dnn-2fef48ea3984. (Diakses 10 November 2021).

Jupiyandi, S., Saniputra, F. R., Pratama, Y., Dharmawan, M. R., & Cholissodin, I. (2019). Pengembangan Deteksi Citra Mobil Untuk Mengetahui Jumlah Tempat Parkir Menggunakan Cuda Dan Modified Yolo Development of Car Image Detection To Find Out the Number of Parking Space Using Cuda and Modified Yolo. *JTIIK*. 6(4): 413-419.

Karimi, G. (2021). *Introduction to YOLO Algorithm for Object Detection*. Diperoleh dari:https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection**.** (Diakses 5 Mei 2021).

Karlina, O. E., Indarti, D. (2019). Pengenalan Objek Makanan Cepat Saji Pada Video dan Real Time Webcam Menggunakan Metode You Look Only Once (YOLO). *SINTA*. 24(3): 199-208.

Morgunov, A. (2021). *Object Detection with YOLO: Hands-on Tutorial*. Diperoleh dari:https://neptune.ai/blog/object-detection-with-yolo-hands-on-tutorial. (Diakses 27 November 2021).

Nasution, Muhammad Harun Alrasyid. (2020). *Deteksi Objek Sekitar Menggunakan Algoritma You Only Look Once (Yolo) Untuk Auntonomous Electric Vehicle*. Skripsi tidak dipublikasikan. Palembang: Universitas Sriwijaya.

Pjreddie. (2018). *YOLO: Real-Time Object Detection*. Diperoleh dari: https://pjreddie.com/darknet/yolo. (Diakses 13 April 2021).

Pramestya, Ravy Hayu. (2018). *Deteksi Dan Klasifikasi Kerusakan Jalan Aspal Menggunakan Metode Yolo Berbasis Citra Digital.* Tesis tidak dipublikasikan. Surabaya: Institut Teknologi Sepuluh November.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). (YOLO) *You Only Look Once*. Cvpr. https://doi.org/10.1109/CVPR.2016.91.

Rosebrock, A. (2018). *YOLO object detection with OpenCV*. Diperoleh dari: https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv. (Diakses 18 Agustus 2021).

Rockikz, A. (2021). *How to Perform YOLO Object Detection using OpenCV and PyTorch in Python*. Diperoleh dari:https://www.thepythoncode.com/article/yolo-object-detection-with-opencv-and-pytorch-in-python. (Diakses 11 November 2021).

Santoso, A., & Ariyanto, G. (2018). Implementasi Deep Learning Berbasis Keras Untuk Pengenalan Wajah. Emitor. 18(1): 15-21.

Sutjiadi, R., & Pattiasina, T. J. (2020). Deteksi Objek menggunakan Dashboard Camera untuk Sistem Peringatan Pencegah Kecelakaan pada Mobil. *JTIIK*. 7(2): 427-434.

Tanuwijaya, Evan., & Fatichah, Chastine. (2020). Penandaan Otomatis Tempat Parkir Menggunakan YOLO untuk Mendeteksi Ketersediaan Tempat Parkir Mobil pada Video CCTV. *Brilliant.* 5(1): 189-198.

Tharhan, Muhammad Rafly. (2020). *Rancang Bangun Aplikasi Deteksi Kerusakan Jalan Menggunakan Algoritma Yolo.* Skripsi tidak dipublikasikan. Depok: Politeknik Negeri Jakarta.

Umam, K., & Negara, B. S. (2016). Deteksi Obyek Manusia Pada Basis Data Video Menggunakan Metode Background Subtraction Dan Operasi Morfologi. *CoreIT.* 2(2): 31-40.