KLASIFIKASI PENYAKIT DAUN ANGGUR MENGGUNAKAN YOLOV4 DAN OPENCV PADA APLIKASI WEB

Randy Ardiansyah[1], Lalu A. Syamsul Irfan A[2], A. Sjamsjiar Rachman[3]

[1,2,3] Jurusan Teknik Elektro Universitas Mataram
[1]randyardiansyah2112@gmail.com, [2]irfan@unram.ac.id, [3]sjamjiar@unram.ac.id

ABSTRAK

Anggur merupakan tanaman yang hidup di dataran rendah, namun seringkali penyakit terlihat pada bentuk daun tanaman anggur yang dapat mempengaruhi hasil produksi tanaman tersebut, untuk mengantisipasi penyebaran penyakit tanaman anggur tersebut diperlukan sebuah model *machine learning* yang dapat melakukan klasifikasi penyakit tanaman anggur tersebut. Penelitian ini melakukan perancangan aplikasi web menggunakan framework flask dalam melakukan klasifikasi terhadap penyakit di daun anggur secara *realtime* melalui webcam dan juga melalui masukan citra dari perangkat ataupun melalui kamera belakang maupun kamera depan jika pada web mobile. Model *machine learning* yang digunakan dalam klasifikasi pada aplikasi web menggunakan model YOLOv4 dari framework darknet dengan jumlah data *training* yang digunakan sebanyak 5414 data citra dan data uji sebanyak 1354 data citra. *Mean Averange Precision* yang di dapatkan model YOLOv4 menyentuh 100% dengan *averange loss* sebanyak 0.2897 dengan kelas yang digunakan yaitu *black measled*, *black rot*, *healthy*, dan *leaf blight* dalam melakukan klasifikasi pada proses pelatihannya.

Kata Kunci: Anggur, Klasifikasi, Daun, YOLOv4

ABSTRACT

Grapes are plants that live in the lowlands, but often diseases are seen in the shape of the leaves of grape plants that can affect the production of these plants, to anticipate the spread of grape vine diseases, a machine learning model is needed that can classify grape vine diseases. This research designs a web application using the flask framework in classifying diseases on grape leaves in realtime through a webcam and also through image input from the device or through the rear camera or front camera if on a mobile web. The machine learning model used in the classification of web applications uses the YOLOv4 model from the darknet framework with the amount of training data used as much as 5414 image data and 1354 test data image data. Mean Averange Precision obtained by the YOLOv4 model touched 100% with averange loss of 0.2897 with the classes used, namely black measled, black rot, healthy, and leaf blight in classifying the training process.

Keywords: Grape, Classification, Leaf, YOLOv4

1. PENDAHULUAN

Anggur adalah salah satu tanaman yang tumbuh dengan cara merambat serta pada rantingnya mengeluarkan buah yang banyak. Tanaman ini termasuk dalam tanaman *perennial* atau tahunan dengan memiliki banyak golongan keluarga *Vitacaee*. Buah anggur memiliki kandungan senyawa yang bagus untuk kesehatan. Masalah yang sering ada pada tanaman anggur sendiri adalah penurunan jumlah mutu produksi anggur yang disebabkan mikroorganisme serta jamur yang dapat mengakibatkan penyakit. Daun anggur yang terkena penyakit tersebut dapat dilihat langsung namun kemiripan penyakit yang menyerang daun anggur tersebut sulit dibedakan karena warna serta tekstur yang ada pada daun.

Pengendalian serta pengetahuan tentang penyakit pada tanaman anggur khusunya pada daun merupakan langkah penting yang dapat memberi pengaruh dari produksi buah anggur. Penyakit yang menyerang daun anggur antara lain *Black Rot*, *Esca* atau *Black Measlead*, dan *Leaf Blight*. Pengendalian jamur dan mikroorganisme yang menyebabkan penyakit tersebut di Indonesia masih menggunakan cara yang tradisional. Cara yang biasa dilakukan oleh petani adalah memberikan pestisida kepada tanaman tersebut, namun pemberian pestisida dapat memberikan dampak negatif bagi lingkungan sekitar serta kesehatan. Kebutuhan petani terhadap pakar penyakit tanaman sangat diperlukan agar dapat memberikan analisa yang akurat terhadap penyakit yang di diderita tanaman tersebut, terutama pada tanaman anggur ini. Namun orang yang ahli dibidang tersebut masih kurang, oleh sebab itu salah satu cara yang dapat dilakukan oleh para petani adalah memanfaatkan teknologi yang dapat membantu dalam pengendalian penyakit daun anggur tersebut. Perkembangan teknologi dalam bidang *machine learning* terutama *deep learning* dapat membuat komputer mampu mendeteksi penyakit pada daun anggur dengan cara memberikan pelatihan kepada model *machine learning* tersebut.

Berbegai penelitian telah dilakukan dalam melakukan deteksi terhadap penyakit yang tampak pada daun anggur tersebut, Ghoury, dkk. (2019) melakukan penelitian yang membuat sistem pemrosesan otomatis dalam mendeteksi penyakit yang ada pada anggur menggunakan 2 metode untuk melakukan deteksi pennyakit pada buah anggur dan daun anggur. Metode

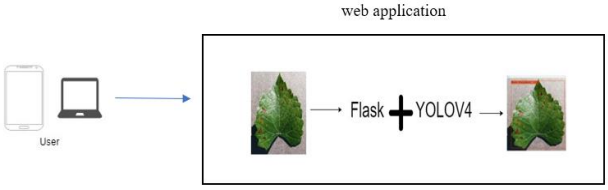
yang digunakan adalah SSD MobileNet v1 dan Faster R-CNN Inception v2. Total data yang digunakan adalah 656 data citra dengan jumlah data citra untuk *training* adalah 543 citra dan untuk *testing* sebesar 113 citra. Jumlah kelas yang digunakan ada 4 yaitu *Healthy Grape* atau HG, *Healthy Grape Leaf* atau (HGL), *Diseased Grape* (DG), *Diseased Grape leaf* (DGL). Model Faster R-CNN Inception v2 mampu mendeteksi serta mengklasifikasikan 95,57% dari data uji dengan akurasi dari 78% sampai 99%, sedangkan untuk model SSD_MobileNet v1 hanya mampu mengklasifikasi data uji sebesar 59,29%, tetapi dengan citra yang mengandung sedikit noise dan citra terorganisir dapat membuat akurasinya mencapai 90% sampai 99%,untuk citra yang mengandung noise yang berbeda beda di latar belakang dapat membuat akurasinya menjadi sangat buruk diantara 52% sampai 80%. Xie, dkk. (2020) melakukan penelitian untuk mendeteksi penyakit pada daun anggur menggunaakan metode Faster DR-IACNN secara *realtime* dengan penyakit yang di deteksi antara lain *Black rot*, *Leaf blight*, *Black measles*, dan *Mites*. Metode ekstraksi fitur yang digunakan sendiri ada beberapa modul antara lain Inception-v1, modul Inception-ResNet-v2, dan modul Inception-ResNet-v2. Data citra yang digunakan dibagi menjadi 60% untuk *training* dan 40 % untuk validasi serta *testing*. Faster DR-IACNN yang digunakan mampu mencapai 81,1% mAP.

Penelitian selanjutnya oleh Zhu, dkk. (2021) memiliki tujuan deteksi *black rot* pada daun anggur dengan jumlah data citra sebesar 1180. Metode yang digunakan adalaha metode YOLOv3-SPP dengan mengganti *Intersection Over Union* atau IOU dalam jaringan YOLOv3 diganti menggunakan *Generalized Intersection Over Union* atau GIOU dan penambahan modul *Spatial Pyramid Pooling* dalam meningkatan deteksi. Akurasi deteksi yang didapatkan adalah 95,79% dan *recall* sebesar 94,52%, di mana akurasi lebih besar 5,94% dan 10,67% *recall* dari YOLOv3 asli. Pada gambar latar belakang sederhana juga di dapatkan akurasi sebesar 94,05%. Guang, dkk. (2023) melakukan penelitian yang bertujuan untuk dapat mendeteksi penyakit secara *realtime* menggunakan metode YOLO-SL dengan dengan jumlah *class* yang digunakan sebanyak 4 *class* dengan 3591 data citra dengan 1180 untuk *black rot*, 1383 *Ringworm*, 400 *Leaf spot*, 628 *Brown spot*. YOLOv3 yang ditingkatkan dengan menggunakan ekstraksi fitur ShuffleNetv2 dalam melakukan pengurangan parameter di jaringan. Percobaan yang dilakukan dengan metode YOLO-SL ini medapatkan akurasi deteksi rata-rata 90,4% dengan waktu deteksi dikurangi dengan rata rata 32,2 ms serta bobot YOLOv3 asli yang dikurangi sekitar 18,3 %.

Berdasarkan penelitian sebelumnya dalam berbagai metode deteksi kami mengusulkan dalam pembuatan aplikasi berbasis web yang dapat klasifkasi penyakit pada daun anggur dengan menggunakan YOLOv4 dan OpenCV, dengan fitur yang yang ada aplikasi tersebut antara lain *input* gambar citra daun anggur, menangkap gambar secara langsung dari kamera belakang maupun depan, dan deteksi secara *realtime* lewat webcam pada laptop terhadap penyakit daun anggur tersebut.

2. METODE PENELITIAN

Penelitian sebelumnya telah melakukan berbagai metode deteksi dalam meleakukan deteksi penyakit pada daun anggur, pada penelitian ini dilakukan penggabungan model kecerdasan buatan dalam aplikasi berbasis web dalam melakukan deteksi dan klasifikasi penyakit pada daun anggur secara realtime. Adapun cara kerja aplikasi tersebut secara sederhana dapat ditunjukan pada Gambar 1.



Gambar 1 Cara Kerja Sistem

Cara kerja aplikasi berbasis web pada gambar 1 adalah pengguna dapat melakukan input gambar daun anggur untuk di deteksi dan klasifikasi ataupun dapat menangkap gambarnya menggunakan kamera perangkat yang digunakan jika di web mobile dan secara realtime melalui webcam laptop.

a. Studi Literatur

Pada studi literatur peneliti mengumpulkan informasi dari literatur-literur yang berkaitan mengenai penyakit yang ada pada daun anggur, proses pengembangan web dengan *framework* flask, dan cara untuk menggunakan model YOLOv4 di website. Literaturnya sendiri dapat didapatkan dari buku, jurnal, dan referensi yang lain.

b. Pengumpulan Data

Data yang akan digunakan pada penelitian ini sendiri adalah data citra daun anggur. Data data tersebut didapatkan dari website kaggle dari akun samir bhattarai dengan judul *new plant disease dataset*. Dataset tersebut memiliki banyak citra daun yang ada penyakit, terutama citra daun anggur. Data citra daun anggur yang ada akan dibagi menjadi 4 *class*. Penulis menggunakan data *train* citra daun anggur tersebut untuk menjadi data *testing* dan data *train*. Jumlah data citra daun anggur dari 4 *class* tersebut pada *train* adalah 7222 citra, tetapi supaya setiap data citra dari setiap *class* seimbang maka penulis menggunakan 1692 citra dari setiap *class* sehingga total data citra yang digunakan adalah 6768 data citra. Pembagian dataset nanti akan dilakukan dengan pembagian untuk data *training* 80% dan *testing* 20%. Pembagiannya dapat dilihat di tabel 1 sebagai berikut :

Tabel 1 Pembagian Dataset Citra

No	Dataset	Rasio	Jumlah
1	Data *training*	80%	5414
2	Data *testing*	20%	1354

Pada Tabel 1 data pelatihan yang berjumlah 5414 data citra yang terbagi menjadi *black measled* sebanyak 1370 citra, *black rot* sebanyak 1361 citra, *healthy* sebanyak 1342 citra, dan *leaf blight* sebanyak 1341 citra. Data *testing* yang digunakan juga terbagi menjadi *black*

measled sebanyak 322 citra, *black rot* sebanyak 331 citra, *healthy* sebanyak 350 citra, dan *leaf blight* sebanyak 351 citra. Adapun contoh dataset ditunjukkan pada Gambar 2.

Gambar 2 Contoh dataset

c. Pengubahan nama file gambar pada dataset

Pengubahan nama file gambar dari dataset ini bertujuan untuk mempermudah dalam melakukan anotasi setiap citra dalam menentukan *class*nya. Pengubahan dilakukan setelah dataset dibagi menjadi data yang seimbang di setiap *class* dan dilakukan di windows. Setelah perubahan nama selesai, data citra akan digabungkan dalam 1 folder dan akan dihilangkan spasi pada setiap data citra, hal ini supaya dapat mempermudah dalam pengujian.

d. Pemberian anotasi pada dataset

Setelah pengubahan nama file citra dari setiap citra tersebut, penulis melakukan pemberian anotasi dengan menggunakan yolo mark. Yolo mark sendiri adalah salah satu tools yang dapat digunakan untuk memberikan anotasi citra dalam melakukan *training* terhadap YOLOv4. *Tools* yolo mark ini menyedikan file file untuk dapat melakukan *training* untuk YOLOv4, file tersebut meliputi obj.data, obj.names, dan file img. File obj.data sendiri berisi pengaturan jumlah *class* yang digunakan dalam anotasi, pengaturan file *backup* dan pengaturan penyimpanan data *train* dan validasi berupa data testing. Adapun contoh konfigyrasinya dapat dilihat pada gambar 3.

```
classes= 4
train  = data/train.txt
valid  = data/test.txt
names = data/obj.names
backup = backup/
```

Gambar 3 Konfigurasi file obj.data

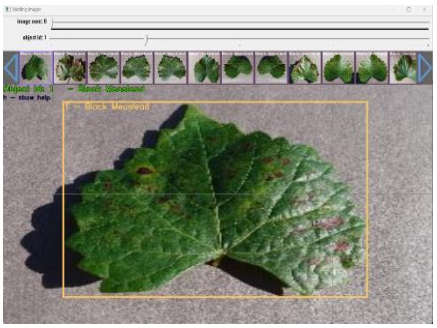
Terdapat juga file obj.names adalah file yang digunakan memberi nama *class* untuk dilakukan anotasi, dan file img untuk data menyimpan data gambar. Adapun isi dari file obj.names ditunjukkan pada gambar 4

```
File    Edit    View

Black Rot
Black Measlead
Healthy
Leaf Blight
```

Gambar 4 isi file obj. names

Pemberian anotasi pada dataset citra daun anggur dengan yolo mark ditunjukan pada Gambar 5 setelah pengaturan konfigurasi yang lain telas selesai.



Gambar 5 Contoh Proses Anotasi

Hasil dari pemberian *bounding box* ini pada citra tersebut adalah sebuah file txt dengan nama yang sama dengan nama citra tersebut. Adapun contoh isi file txt tersebut dapat dilihat pada Gambar 4.4.

```
File    Edit    View

1 0.493750 0.480556 0.707812 0.816667
```

Gambar 6 Contoh Isi File Txt Hasil Pelabelan

Pada Gambar 6 terdapat beberapa angka hasil pelabelan dari citra. Angka angka tersebut dapat dijelaskan sebagai berikut :

- Angka 1 adalah id class yang digunakan pada pelabelan.
- 0.493750 adalah X_CENTER yang merupakan koordinat pusat pada x untuk dideteksi.
- 0.480556 adalah Y_CENTER yang merupakan koordinat pusat pada y untuk dideteksi.
- 0.707812 adalah lebar dari objek yang akan dideteksi.
- 0.816667 adalah tinggi dari objek yang akan dideteksi.

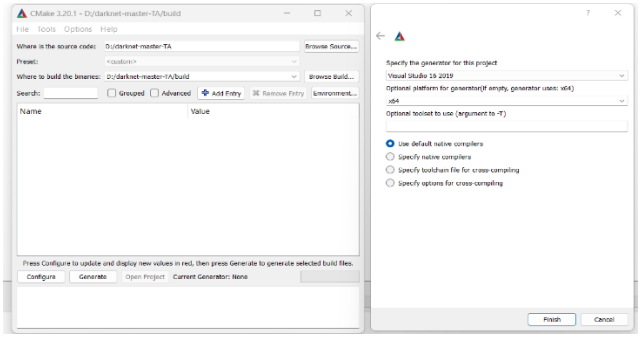
e. Instalasi YOLOV4

Proses instalasi YOLOv4 pada penelitian ini menggunakan mode GPU yang berbasis NVDIA. Instalasi YOLOv4 ini sendiri dilakukan dengan setelah menginstal driverGPU, CUDA, cuDNN, dan melakukan *build* Darknet dengan mendownload di repository github https://github.com/pHidayatullah/darknet Proses selanjutnya adalah melakukan ektrask file yang sudah di *download* dan melakukan konfigurasi di file CmakelList.txt.

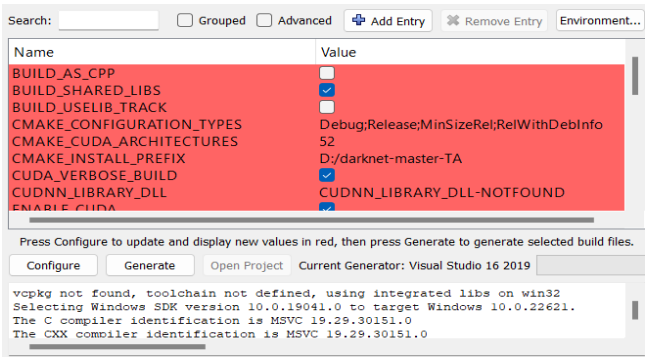
```
option(CMAKE_VERBOSE_MAKEFILE "Create verbose makefile" ON)
option(CUDA_VERBOSE_BUILD "Create verbose CUDA build" ON)
option(BUILD_SHARED_LIBS "Create dark as a shared library" ON)
option(BUILD_AS_CPP "Build Darknet using C++ compiler also for C files" OFF)
option(BUILD_USELIB_TRACK "Build uselib_track" ON)
option(MANUALLY_EXPORT_TRACK_OPTFLOW "Manually export the TRACK_OPTFLOW=1 define" OFF)
option(ENABLE_OPENCV "Enable OpenCV integration" ON)
option(ENABLE_CUDA "Enable CUDA support" ON)
option(ENABLE_CUDNN "Enable CUDNN" ON)
option(ENABLE_CUDNN_HALF "Enable CUDNN Half precision" OFF)
option(ENABLE_ZED_CAMERA "Enable ZED Camera support" ON)
option(ENABLE_VCPKG_INTEGRATION "Enable VCPKG integration" ON)
option(VCPKG_BUILD_OPENCV_WITH_CUDA "Build OpenCV with CUDA extension integration" ON)
```

Gambar 7 Konfigurasi file CmakeList.txt

Kemudian akan dilakukan *configure* dan generate menggunakan Cmake(GPU) yang ditunujukan pada Gambar 8 dan juga Gambar 9.

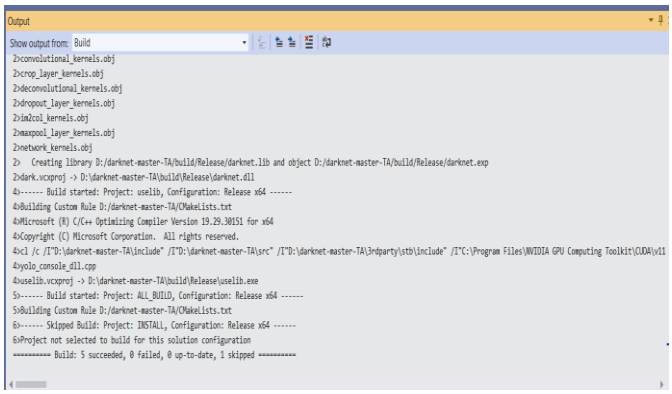


Gambar 8 Proses Sebelum *Configure* Folder *Darknet*



Gambar 9 Hasil *Configure* Folder *Darknet*

Proses selanjutnya setelah *configure* adalah menekan tombol *Generate* telah ada dan selanjutnya menekan tombol *Open Project* untuk membuat projek pada visual studio 2019 community, saat terbuka maka perlu dilakukan perubahan mode dari mode *Debug* ke mode *Release* hingga menekan tombol *Build* dan menakan tombol *build solution*. Adapun *output* dari proses *build* dapat dilihat pada Gambar 10.



Gambar 10 *Output* Hasil Proses *Build*

Proses terakhir adalah melakukan konfigurasi file yang ada pada folder darknet yang telah di proses. Prosesnya adalah sebagai berikut :

1. Melakukan *copy* folder data dan cfg yang berada pada folder darknet-master-TA, hasil copy lalu di paste di darknet-master-TA/build/Release.
2. Melakukan *copy* file pthreadVC2.dll dari folder darknet-master-TA/3rdparty/pthreads/bin, hasil *copy* file di *paste* di darknet-master-TA/build/Release.

Setelah proses tersebut selesai maka YOLOv4 dapat digunakan di windows dengan menggunakan framework Darknet.

f. Pelatihan model YOLOV4

Pelatihan model YOLOv4 adalah tahapan untuk melatih model untuk dapat mendeteksi serta klasifikasi penyakit daun anggur. Pelatihan model YOLOv4 ini dilakukan setalah dilakukan anotasi pada citra. Sebelum melakukan proses pelatihan model, maka perlu melakukan proses pembagian data citra menjadi *trainig* dan *testing* serta konfigurasi hyperparameter untuk YOLOv4. Konfigurasi ini dilakukan pada file cfg dengan pengaturan *batch*, *subdivision*, *max batchs*, *steps* dari *max batch*, *size input* untuk YOLOv4, jumlah *class* untuk setiap *layer* YOLO, jumlah *filter* untuk setiap *convolutional* sebelum *layer* YOLO. Adapun pengaturan tersebut dapat dilihat pada Tabel 1 dan Tabel 2.

Tabel 1 Konfigurasi sebelum layer convolutional

No	Hyperparameter	Nilai
1	subdivision	64
2	*max batchs*	8000
3	*steps* dari *max batch*	6400, 7200
4	*size input* (*width dan hight*)	224
5	*batch*	64

Tabel 2 Konfigurasi layer convolutional sebelum layer Yolo dan layer Yolo

No	Hyperparameter	Nilai
1	*Filter*	27
2	*Classes* (Jumlah kelas)	4

Setelah proses hyperparameter dilakukan maka proses *training* dapat dilakukan dengan masuk ke folder *build* YOLOv4 lalu membuka terminal dan mengetikan perintah seperti berikut:

```
darknet.exe detector train data/obj.data cfg/yolov4-obj.cfg yolov4.conv.137 – map
```

g. Evaluasi Model

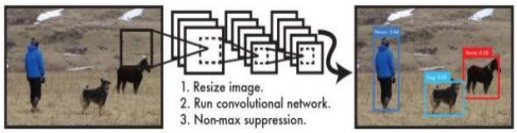
Evaluasi model adalah proses yang dilakukan untuk menguji seberapa baik model dalam melakukan deteksi dan klasifikasi terhadap data yang belum pernah dilihat oleh model. Pada YOLOv4 saat proses *training* selesai maka akan terdapat chart yang menggambar hasil dari proses *training* dalam setiap iterasi pada saat pelatihan. Pada chart tersebut juga menggambarkan *avg loss* pada saat pelatihan serta mAP yang didapatkan. Evaluasi pada penelitian ini juga akan menggunakan *confusion matrix* untuk mengevaluasi model yang dihasilkan.

h. Pengembangan perangkat lunak

Proses pada pengembangan perangkat lunak ini adalah melakukan analisa sistem, perancangan sistem, pengembangan sistem, dan pengujian sistem dengan mengimplementasikan Model YOLOv4 kedalam aplikasi berbasis web dengan menggunakan OpenCV dan juga flask.

1. YOLOv4

You Only Look Once atau YOLO merupakan metode untuk melakukan deteksi objek yang menggunakan single CNN dengan memprediksi probabilitas dan *bounding box class* dalam satu kali evaluasi.



Gambar 11 Deteksi YOLOv4
(Kusuma, dkk. 2021)

Pada Gambar 11 setalah citra *input* di dapatkan maka sistem mengubah ukuran citra dengan ukuran 416 x 416 dan kemudian akan di proses oleh *single* CNN. Proses selanjutnya adalah sistem akan melakukan *non-max suppression* dalam mendapatkan *bounding box* untuk kelas tiap objek. YOLOv4 adalah pengembangan dari YOLOV3, Arsitektur yang digunakan pada YOLOV4 adalah darknet-53 yang memiliki peran sebagai pengakstraksi citra pada input dengan *backbone* yang digunakan adalah CSPDarknet-53 yang sering disebut CSPNet dengan *capability learning* yang ditingkatkan dari CNN (Kusuma, dkk. 2021). Namun dalam beberapa percobaan yang telah dilakukan CSPDarknet53 menjadi *backbone* yang optimal dari pada *backbon*e lainnya (Hidayatullah, 2021).

Backbone sendiri adalah arsitektur dalam melakukan ekstraksi fitur, selain CSPDarknet53 ada juga *backbone* yang lain seperti CSPReNext50 dan Efficient-B3. Pengaruh dari kemampuan yang ada di CSPDarknet53 tersebut pada saat dilakukan *input* citra adalah meningkatkan total kombinasi di setiap titik pada suatu citra dan dapat melihat objek menjadi semakin detail. SPP yang ada pada neck berfungsi untuk tidak menurunkan kecepatan pada operasi jaringan serta memisahkan secara signifikasn fitur konteks (Rahman, dkk. 2022). head yang merupakan tempat dilakukan prediksi untuk membuat *bounding box*. Adapun rumusnya untuk menentukan *bounding box* tersebut adalah sebagai berikut:

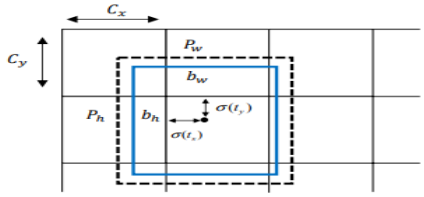
$$b_x = \sigma(t_x) + C_x \quad (1)$$

$$b_y = \sigma(t_y) + C_y \quad (2)$$

$$b_w = P_w e^{t_w} \quad (3)$$

$$b_h = P_h e^{t_h} \quad (4)$$

Dimana C_x dan C_y mewakili *offset* grid relatif terhadap kiri atas, P_w dan P_h adalah lebar dan tinggi anchor, b_x mengambil titik tengah kotak pembatas standar prediksi, b_w dan b_h adalah lebar serta tinggi frame yang diprediksi masing masing, dan σ adalah fungsi sigmoid. t_x, t_y,t_w,t_h mewakili nilai prediksi jaringan, dengan menggunakan parameter ini YOLOv4 menyelesaikan penyesuaian frame (Zakria, dkk, 2022). Ilustrasi persamaan 1 sampai 4 dapat dilihat pada Gambar 12.



Gambar 12 *Bounding Box YOLOv4 Prediksi*
(Fu, dkk, 2021)

Bagian head ini juga tempat dilakukan perhitungan *confidance* dan juga *loss* pada proses *training* berlangsung, *loss* sendiri pada YOLOv4 ada tiga yaitu *loss* posisi *bounding box*, *loss confidance* dan *loss* untuk kategori *class*nya. Adapun persamaan untuk *confidance* ditunjukan pada persamaan 5 serta *loss function* ditunjukkan pada persamaan 6 sampai dengan 9.

Confidance= $P_r(Class_i|Object)$ x $P_r(Object)$ x IoU_{pred}^{truth} (5)

$$Loss = Loss_1 + Loss_2 + Loss_3 \quad (6)$$

$$Loss_1 = (1 - CIOU) \quad (7)$$

$$CIOU = IOU - \frac{\rho^2(b,b^{gt})}{c^2} - \beta v$$

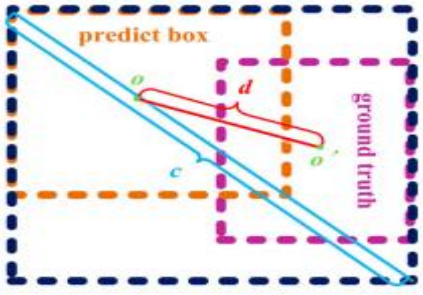
$$\beta = \frac{v}{1 - IOU + v}$$

$$v = \frac{4}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h})^2$$

$$Loss_2 = \sum_{i=0}^{s^2}\sum_{j=0}^{B} W_{ij}^{obj}[\hat{C}_i^j \log(C_i^j) + (1 - \hat{C}_i^j)\log(1 - C_i^j)] - \lambda_{noobj}\sum_{i=0}^{s^2}\sum_{j=0}^{B}(1 - W_{ij}^{obj})[\hat{C}_i^j \log(C_i^j) + (1 - \hat{C}_i^j)\log(1 - C_i^j)] \quad (8)$$

$$Loss_3 = -\sum_i^{S^2}\sum_j^B W_{ij}^{obj}\sum_{C=1}^{C}[\hat{p}_i^j(c)\log\left(p_i^j(c)\right) + \left(1 - \hat{p}_i^j(c)\right)\log(1 - p_i^j(c))] \quad (9)$$

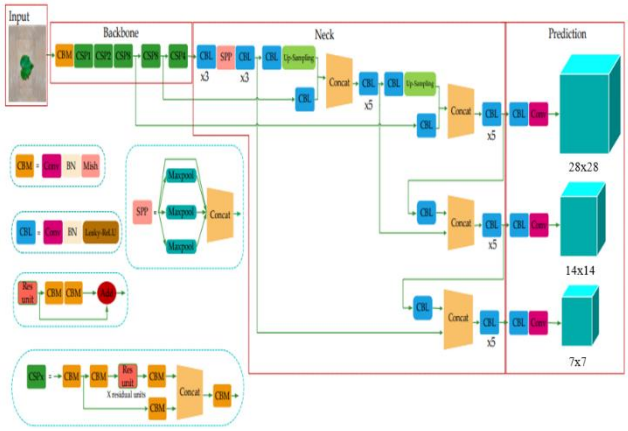
Persamaan 5 pada $P_r(Class_i|Object)$ menunjukan probabilitas jenis objek yang diketahui sebagai suatu benda. $P_r(Object)$ sendiri mewakili probabilitas jika kotak prediksi berisi objek, jika ada maka $P_r(Object)$ =1 sedangkan jika tidak ada maka 0 (Yu, dkk, 2021). *Loss* pada YOLOv4 dihitung dengan menjumlahkan *loss bounding box* regresi ($Loss_1$), *loss confidance* ($Loss_2$) dan *loss* klasifikasi ($Loss_3$) seperti persamaan 6. Pada persamaan 7 merupakan persaman untuk menghitung *loss bounding box* dimana terdapat IoU (*Intersection over union*), w^{gt} serta h^{gt} adalah lebar dan tinggi dari *ground truth bounding box* pada saat anotasi, w serta h adalah prediksi lebar dan tingi dari *bounding box* prediksi, $\rho^2(b, b^{gt})$ merupakan jarak euclidean antara titik pusat *bounding box* prediksi dan *bounding box ground truth*, dan c adalah jarak minimal diagonal kotak yang memuat kotak pembatas prediksi dan *bounding box ground truth* (Jiang, dkk, 2020). Adapun ilustrasi *bounding box* dengan *ground truth* dapat dilihat pada gambar 13.



Gambar 13 ilustrasi *bounding box* dan *ground truth*
(Gai, dkk, 2021)

Persamaan 8 memiliki s^2 yang merupakan nomer *grid input* gambar dan B adalah nomer *bounding box* yang ada di dalam *grid*, W_{ij}^{obj} hanyalah sebuah fungsi objek dimana jika *bounding box* ke j dalam *grid* ke i dideteksi

objek maka $W_{ij}^{obj} = 1$ apabila tidak maka nilainya 0. $\hat{C}_i^j$ dan C_i^j adalah *confidance score* prediksi dan *confidance score ground truth* pada *bounding box* ke j pada *grid* ke I, sementara λ_{noobj} adalah bobot parameter. Persamaan 9 memiliki $p_i^j(c)$ dan $\hat{p}_i^j(c)$ adalah prediksi probabilitas dan probablitas *ground truth* dari objek c untuk diklasifikasikan pada *bounding box* ke j pada *grid* ke I (Jiang, dkk, 2021). Adapun arsitektur YOLOv4 sendiri dapat dilihat pada Gambar 14.



Gambar 14 Arsitektur YOLOv4
(Yu, dkk,2021)

Pada Gambar 14 terdapat bagian *input* citra sesuai resolusi yang digunakan pada model YOLOv4, pada penelitian ini menggunakan resolusi 224 x 224 dengan *channel* RGB. Selanjutnya bagian *backbone* merupakan bagian pengekstrak ciri citra dengan menerapakan proses konvolusi beserta fungsi aktivasi, pada backbone sendiri terdapat CBM (*Convolution Batch Normalization Mish*) dan proses CSP. Proses selanjutnya menuju *Neck* yang merupakan lapisan antara *backbone* dan juga *dense prediction* yang tempat dilakukannya SPP atau *Spatial Pyramid Pooling*, sementara di *Dense prediction* merupakan bagian tempat dilakukannya pembentukan *bounding box* dan juga klasifikasi, untuk lebih jelasnya berdasarkan gambar 3.4 maka proses dalam arsitektur YOLOv4 dapat dijelaskan pada Gambar 3.5.

2. OpenCV

OpenCV merupakan perangkat lunak untuk pembelajaran serta memiliki infrastruktur umum pada aplikasi di visi komputer. OpenCV ini juga perpustakaan komputer untuk visi *open source* yang dapat mempercepat proses produk komersial dalam pemakaian persepsi mesin. 2500 Algoritma ada pada openCV ini yang sudah dioptimalkan dengan mencakup algoritma pembelajaran mesin, *computer vision state-of-the-art*, dan paket lengkap untuk komputer visi klasik. *Library* yang ada pada komputer *vision* sudah lengkap ada pada openCV serta merupakan metode yang cepat (Muchtar dan Apriadi, 2019).

3. Flask

Flask merupakan *framework* yang ada di pyhton, flask tidak terlalu memiliki banyak *tools* ataupun *library* sehinga sering disebut sebagai *micro-framework* untuk web. Pengembangan yang berkapasitas kecil pada program serta memori yang digunakan relatif kecil membuat pemakaian sumber daya menjadi kecil adalah salah satu tujuan dari penggunaan flask, sehingga pengembangan lebih efisien. *Framework* flask meskipun ringan tetapi fungsi yang dibutuhkan bisa berjalan. Komponen yang ada flask sendiri ada 2 yaitu Jinja2 dan Werkzeug. Jinja2 digunakan untuk *template engine* dan Werkzeug berfungsi untuk menyedekikan *routing*, *Web Server Interface Gateway* (WSGI), dan *debugging* (Ngantung dan Pakereng, 2021).

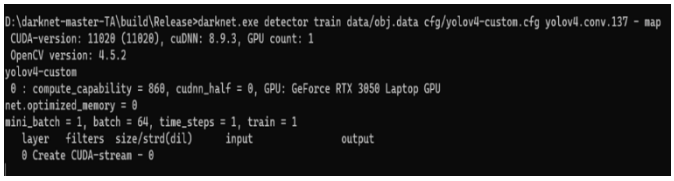
3. HASIL

a. Proses Training

Proses eksekusi *training* model dilakukan dengan masuk ke folder hasil *build* tepatnya di folder release yaitu D:\darknet-master-TA\build\Release, setelah masuk ke foldernya maka selanjutnya masuk ke terminal dan menjalankan perintah berikut :

```
darknet.exe detector train data/obj.data
cfg/yolov4-custom.cfg yolov4.conv.137 -
map
```

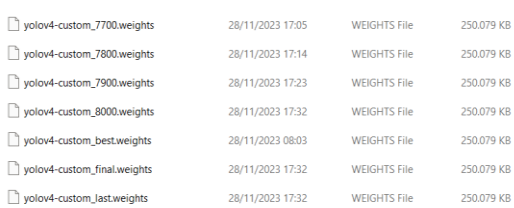
Maksud dari perintah diatas adalah menggunakan darknet.exe sebagai file *executable* YOLOv4 dan *detector train* untuk melakukan *training* model dengan menggunakan file obj.data yang berisi tentang path dataset, jumlah class yang digunakan. Proses *training* juga menggunakan file *config* dengan eksistensi cfg dan *yolov4.conv.137* yang berisi arsitektur serta hyperparameter yang digunakan. Sedangkan perintah terkahir yaitu *-map* untuk bisa menampilkan *char*t hasil pada saat proses *training* nanti. Adapun saat perintah diterminal tersebut dijalankan dapat dilihat pada gambar 15.



Gambar 15 Eksekusi *Training* Model

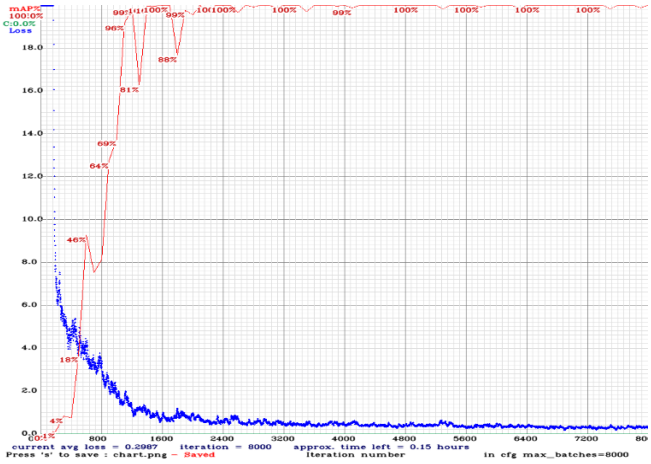
Pada Gambar 15 saat perintah dijalankan maka *darknet* akan melakukan *set up* untuk dapat melakukan *training* dengan mengecek versi CUDA, cuDNN, GPU dan versi OpenCV yang terinstall. Setelah proses itu maka proses *training* akan dilakukan.

Proses *training* yang dilakukan akan menghasilkan *weight* atau bobot dari model yang disimpan pada file *backup*. Bobot yang di simpan sendiri adalah bobot setiap 100 iterasi yang dilakukan, bobot *final*, bobot terakhir, dan bobot yang paling bagus. Isi folder *backup* yang meyimpan bobot dapat dilihat pada Gambar 7.



Gambar 16 Bobot proses *training*

Saat Proses *training* dilakukan dengan perintah *darknet* yang menggunakan *-map* menghasilkan *chart* yang digunakan untuk mengetahui proses pelatihan yang dilakukan oleh model saat iterasi berjalan dengan *treshold* 0.25. *Chart* tersebut dapat dilihat pada Gambar 17.

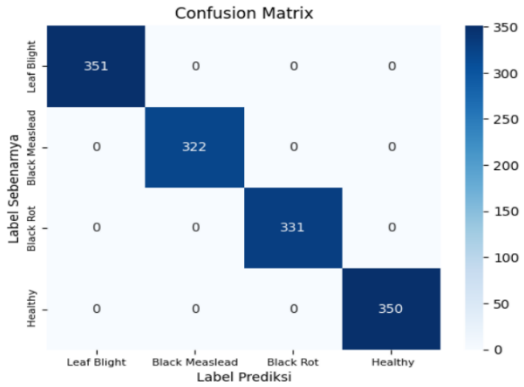


Gambar 17 *chart* proses *training*

Pada Gambar 17 dapat di analisa bahwa pada saat proses *training* dilakukan *averange loss* yang dihasilkan semakin menurun tiap iterasi yang dijalankan. *Averange loss* yang diperoleh adalah 0.2987 dengan jumlah iterasi sebesar 8000 iterasi. Sedangkan mAP yang di dapatkan dari hasil *training* dengan bobot paling bagus adalah 100%, dimana pada setiap iterasi yang dilakukan model berusaha untuk meningkatkan *mean averange precision* yang di dapatkan meskipun di iterasi 700 sampai 1800 masih ada penurunan mAP. Nilai mAP yang 100% yang didapatkan oleh model sendiri diakibatkan pada proses *training* yang dilakukan data uji nya memiliki itensitas cahaya yang bagus dan daun tanaman daun anggur yang cukup dekat,sehingga model dengan cepat dalam deteksi dan klasifikasi.

b. Evaluasi Model

Model yang telah dilatih akan siap untuk dilakukan pengujian dengan data citra yang belum pernah dilihat oleh model untuk dilakuakan deteksi dan juga klasifikasi. Data citra yang akan digunakan untuk pengujian adalah data uji dengan menggunakan *tresh-hold* 0.85 atau dengan kata lain dengan tingkat kepercayaan 85%. Adapun hasil pengujiannya dapat dilihat pada gambar



Gambar 18 *Confusion Matrix* Data Uji

Pada Gambar 9 dapat dilihat *confusion matrix* dari data uji, dengan menggunakan *confusion matrix* tersebut penulis dapat mengetahui performa model dalam melakukan deteksi. Adapun dengan menggunakan confusion matrix tersebut maka kinerja model dapat ditunjukan pada Tabel 3.

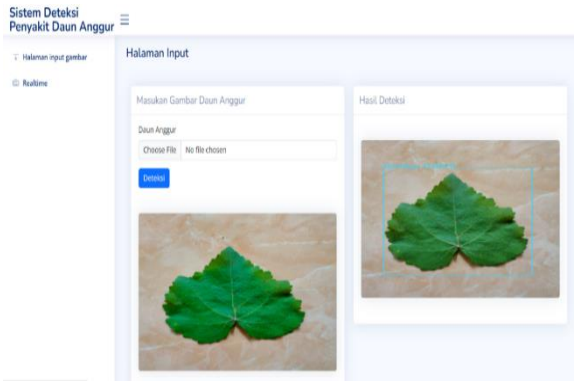
Tabel 3 Hasil Parameter *Confusion Matrix*

Nama *Class*	*Recall*	*Precision*	*FPR*	*F1-Score*
Leaf Blight	1	1	0	1
Black Measlead	1	1	0	1
Black Rot	1	1	0	1
Healthy	1	1	0	1

Pada tabel 3 dapat diketahui bahwa model memiliki nilai *recall* 100% untuk semua *class* yang dilakukan deteksi. Untuk nilai precision yang di dapatkan juga untuk semua *class* mendapatkan nilai *precision* 100%, sedangkan untuk nilai *false positive rate* atau FPR yang di dapatkan oleh model sebesar 0% untuk semua *class*, berarti model tidak memiliki tingkat kesalahan dalam melakukan deteksi dan klasifikasi terhadap citra daun anggur. Terakhir nilai *f1-score* yang di dapatkan untuk semua *class* adalah 100%, dengan nilai *f1-score* merupakan nilai rata rata dari *recall* dan *precision*. Nilai *recall*, *precision* ,dan juga *f1-score* yang 100% diakibatkan pada saat pengujian itensitas cahaya pada data uji yang bagus dan gambar daun anggur cukup dekat membuat model dapat melakukan deteksi dan klasifikasi dengan akurat.

C . Antarmuka Aplikasi Web

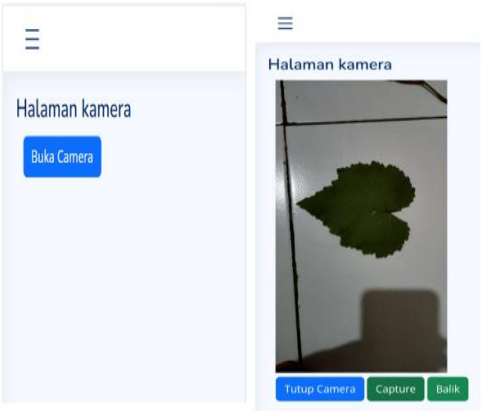
Setelah melakukan evaluasi terhadap model yang telah dibuat maka penulis merancang aplikasi berbasis web dengan menggunakan bobot YOLOv4 yang telah di dapatkan dari hasil pelatihan pada bobot 8000 iterasi. Model di load menggunakan openCV sehingga bisa diintegrasikan pada aplikasi berbasis web. Adapun tampilan layar untuk aplikasi ini dapat ditunjukkan pada gambar 10 sampai gamb



Gambar 19 Halaman *Input*

Pada gambar 19 Adalah halaman awal pada aplikasi untuk melakukan deteksi dan klasifikasi penyakit daun anggur. Tombol deteksi berfungsi untuk melakukan deteksi terhadap citra daun anggur yang telah di uploud.

Setelah tombol ditekan maka citra hasil deteksi dan sebelum deteksi akan di tampilkan.

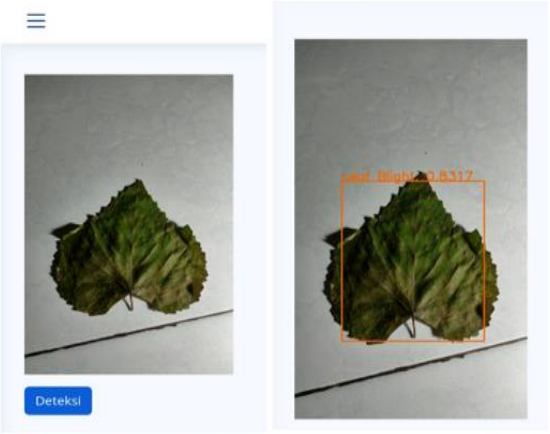


Gambar 20 Halaman foto langsung

Pada gambar 20 merupakan tampilan untuk halaman kamera yang di buka pada web *mobile*, dengan menekan tombol buka kamera maka akan membuka kamera pada ponsel. Adapun penjelasan tombol setelah ditekannya tombol buka kamera adalah sebagai berikut :

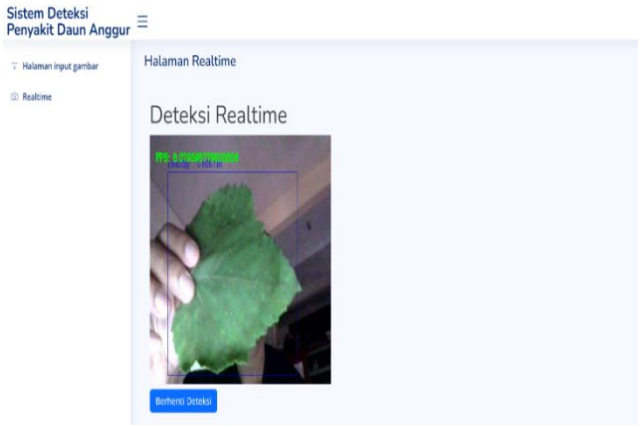
1. Tombol tutup kamera untuk menutup kamera yang terbuka baik kamera depan atau kamera belakang dan berakhir melakukan deteksi.
2. Tombol *capture* adalah tombol yang digunakan untuk menangkap gambar yang ada pada kamera.
3. Tombol balik adalah tombol untuk melakukan perubahan kamera yang ada untuk kamera depan atau kamera belakang.

Setelah menekan tombol *capture* maka gambar akan terambil dan deteksi dapat dilakukan.



Gambar 21 Contoh Hasil deteksi pada halaman foto langsung

Gambar 21 merupakan gambar yang akan dideteksi dan hasil deteksi, saat tombol deteksi ditekan maka akan menghasilkan gambar yang sudah dideteksi dan untuk memulai ulang deteksi dapat menekan tombol *capture* lagi dan menekan tombol deteksi. Untuk halaman terakhir dari aplikasi berbasis web ini adalah halaman *realtime* webcam yang hanya bisa di gunakan melalui laptop pada webcam dapat dilihat pada gambar 4.22.



Gambar 22 Halaman Deteksi *Realtime*

Pada gambar 22 adalah tampilan untuk deteksi *realtime* untuk melakukan deteksi dan klasifikasi penyakit yang ada pada daun anggur. Terdapat tombol berhenti deteksi yang berguna untuk menghentikan deteksi yang di lakukan dan akan menampilkan tombol mulai *realtime* untuk melakukan deteksi lagi.

4. KESIMPULAN

a. Hasil evaluasi pada pengujian model YOLOv4 dari *framework darknet* dengan jumlah kelas yang digunakan ada 4 yaitu *black rot*, *black measlead*, *leaf blight*, dan *healthy* didapatkan nilai *precision*, *recall*, *F1-score* sebesar 100% serta *false positif rate* sebesar 0%.

b. Model yang didapatkan pada proses *training* memiliki kualitas sangat baik dalam melukan klasifikasi penyakit daun anggur sesuai hasil mAP, *precision*, *recall*, *F1-score,* dan *false positif rate* yang didapatkan sehingga dapat digunakan untuk memprediksi penyakit daun anggur.

c. Model YOLOv4 dapat dijalankan pada aplikasi berbasis web dengan bobot pada iterasi 8000 di *load* menggunakan OpenCV sehingga dapat digunakan pada aplikasi berbasis web dalam melakukan prediksi terhadap penyakit pada daun anggur baik secara *realtime* ataupun melalui *input* citra.

DAFTAR PUSTAKA

Fu, H., Song, G., & Wang, Y. (2021). Improved YOLOv4 marine target detection combined with CBAM. *Symmetry*, *13*(4), 623.

Ghoury, S., Sungur, C., & Durdu, A. (2019, April). Real-time diseases detection of grape and grape leaves using faster r-cnn and ssd mobilenet architectures. In *International conference on advanced technologies, computer engineering and science (ICATCES 2019)* (pp. 39-44).

Guang, L. I. U., Guoyu, H. U., ER, G. L. B. H., Tengfei, Z. H. A. O., & Yalan, D. O. N. G. (2023). Detection of grape leaf diseases and insect pests based on improved YOLOv3. Microelectronics and Computers, *40*(2), 110-119.

Hidayatullah, Priyanto. (2021). Buku Sakti Deep Learning: Computer Vision Menggunakan YOLO Untuk Pemula. Stunning Vision AI Academy.

Jiang, Z., Zhao, L., Li, S., & Jia, Y. (2020). Real-time object detection method for embedded devices. In *computer vision and pattern recognition* (Vol. 3, pp. 1-11).

Kusuma, T. A. A. H., Usman, K., & Saidah, S. (2021). People Counting for Public Transportations Using You Only Look Once Method. *Jurnal Teknik Informatika (Jutif)*, *2*(1), 57-66.

Muchtar, H., & Apriadi, R. (2019). Implementasi Pengenalan Wajah Pada Sistem Penguncian Rumah Dengan Metode Template Matching Menggunakan Open Source Computer Vision Library (Opencv). RESISTOR (Elektronika Kendali Telekomunikasi Tenaga Listrik Komputer), 2(1), 39-42.

Ngantung, R. K., & Pakereng, M. I. (2021). Model Pengembangan Sistem Informasi Akademik Berbasis User Centered Design Menerapkan Framework Flask Python. Jurnal Media Informatika Budidarma, 5(3), 1052-1062.

Xie, X., Ma, Y., Liu, B., He, J., Li, S., & Wang, H. (2020). A deep-learning-based real-time detector for grape leaf diseases using improved convolutional neural networks. *Frontiers in plant science*, *11*, 751.

Yu, J., & Zhang, W. (2021). Face mask wearing detection algorithm based on improved YOLO-v4. *Sensors*, *21*(9), 3263.

Zakria, Z., Deng, J., Kumar, R., Khokhar, M. S., Cai, J., & Kumar, J. (2022). Multiscale and direction target detecting in remote sensing images via modified YOLO-v4. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *15*, 1039-1048.

Zhu, J., Cheng, M., Wang, Q., Yuan, H., & Cai, Z. (2021). Grape leaf black rot detection based on super-resolution image enhancement and deep learning. *Frontiers in plant science*, *12*, 695749.