Deteksi Social Distancing Dan Penggunaan Masker Di Restoran Menggunakan Algoritma *Residual Network* (RESNET)

1st Fauzi Bayu Saputra
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
fauzibayus@student.telkomuniversity.ac.id

2nd Meta Kallista
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
metakallista@telkomuniversity.ac.id

3rd Casi Setianingsih
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
setiacasi@telkomuniversity.ac.id

Abstrak*—Penerapan protokol kesehatan social distancing dan penggunaan masker sangat diperlukan karena dampak dari kasus Covid-19 yang semakin meluas. Karena itu cara terbaik dan efektif guna memutus rantai penyebaran Covid-19 salah satunya dengan menerapkan social distancing dan penggunaan masker. Tugas akhir ini akan membahas tentang perancangan dan implementasi deteksi social distancing dan penggunaan masker. Pada pendeteksian ini akan mengambil gambar berdasarkan gambar yang tertangkap oleh kamera kemudian akan dianalisis apakah social distancing dan penggunaan masker diterapkan atau tidak. Deteksi social distancing dan penggunaan masker ini dilakukan secara realtime. Metode You Only Look Once (YOLO) digunakan untuk mendeteksi objek manusia dan metode Residual Network (RESNET) digunakan untuk mendeteksi penggunaan masker, dan menggunakan metode Euclidean Distance untuk mengukur jarak antar objek manusia yang terdeteksi. Berdasarkan hasil yang terbaik dari pengujian dan pembuatan kedua model yang akan digunakan didapat dari rasio dataset yaitu 90% data train dan 10% data test. Dengan hasil pengujian deteksi penggunaan masker akurasi yang didapatkan sebesar 99.04%, dan hasil pengujian deteksi social distancing mAP yang didapatkan sebesar 49.50%.

Kata kunci*—social distancing, penggunaan masker, Covid-19, YOLO, RESNET.

I. PENDAHULUAN

Pandemi *Covid-19* yang sedang berlangsung didunia saat ini merupakan sebuah virus yang menyerang kekebalan tubuh dan akan menyebabkan sindrom pernafasan akut. Asal mula virus corona diketahui pertama kali muncul di pasar hewan dan makanan laut di kota Wuhan, China pada akhir desember 2019 lalu. Dilaporkan kemudian bahwa banyak pasien yang menderita virus ini dan ternyata terkait dengan pasar hewan dan makanan laut tersebut. Orang pertama yang jatuh sakit akibat virus ini juga diketahui merupakan para pedagang di pasar itu. Penyebaran *Covid-19* di Indonesia, Pemerintah secara resmi mengumumkan kasus *Covid-19* pertama di Indonesia pada tanggal 2 maret 2020. Penyebaran virus corona di Indonesia ini tersebar di 34 provinsi di Indonesia. Per tanggal 10 oktober 2021 dari *Our World in Data* dan JHU CSSE *Covid-19* Data, di Indonesia dengan jumlah 4.23 juta kasus dan 143 ribu kasus meninggal dunia [1].

Sehubung dengan adanya pandemi *Covid-19* ini, Pemerintah mengambil langkah untuk melakukan *lockdown* pada kota-kota besar yang mengalami dampak dari *Covid-19* yang menyebar sangat cepat. Setelah melakukan *lockdown*, Pemerintah berinisiatif untuk melonggarkan dari *lockdown* menjadi *social distancing,* yang harus dipatuhi setiap orang harus berjarak kurang lebih satu hingga dua meter, social distancing ini dihimbau pemerintah untuk dipatuhi supaya rantai penyebarannnya tidak semakin luas, dengan menerapkan peraturan 3M yaitu (Memakai masker, Mencuci tangan dan Menjaga jarak)[2].

Pendeteksian tentang social distancing ini sudah pernah dilakukan oleh beberapa peneliti salah satunya yang saya dapatkan, "*Monitoring Social Distancing for Covid-19 Using* OpenCV *and Deep Learning*". Pada penelitian tersebut dilakukan dan dicoba pada luar ruangan dilakukan dengan metode pengawasan yang menggunakan OpenCV, *Computer vision* dan *Deep learning* untuk melacak pejalan kaki dan menghindari kepadatan pejalan kaki. Implementasinya dapat dilakukan dengan menggunakan *Closed Circuit Television* (CCTV) dimana kamera akan mendeteksi keramaian dengan bantuan pendeteksian objek dan menghitung jarak antara keduanya [3].

Pendeteksian penggunaan masker ini sudah pernah dilakukan oleh beberapa peneliti salah satunya yang saya dapatkan, "*Deep Neural Architecture for Face mask Detection on Simulated Masked Face Dataset against Covid-19 Pandemic*". Pada penelitian tersebut dilakukan untuk

mendeteksi yang menggunakan masker dan tidak menggunakan masker dan dicoba menggunakan kamera webcam pada laptop dengan menggunakan model CNN dan VGG16 untuk mengidentifikasinya di antara kerumunan, yang sudah memakai masker dan belum memakai masker [4].

Pada penelitian ini berfokus untuk mengembangkan dari yang sudah ada sebelumnya, yaitu menggabungkan pendeteksian *social distancing* untuk didalam ruangan dengan penambahan pendeteksian penggunaan masker sekaligus. Untuk *social distancing* dengan menggunakan salah satu metode *deep learning You Look Only Once* (YOLO), cara kerja YOLO yang cukup mudah dipahami karena melalui *convulutional network* hanya sekali saja. Membuat YOLO ini sebagai salah satu metode algoritma yang tingkat performansi dan akurasinya cukup optimal [5]. Untuk masker dengan menggunakan salah satu metode deep learning Residual Network (ResNet), cara kerja ResNet yaitu melakukan 50 conv layer untuk mendapatkan hasil yang maksimal. Hal ini membuat ResNet juga termasuk metode algoritma yang memiliki tingkat performansi dan akurasinya cukup optimal [6].

II. KAJIAN TEORI

A. *Social Distancing*

Social Distancing merupakan salah satu langkah untuk mencegah penyebaran Covid-19, dengan upaya menjaga jarak satu sama lain minimal 1-2 meter, menghindari kontak langsung, tidak bersalaman, penundaan acara-acara besar dan lain lain. Penerapan *Social Distancing* ini diharapkan agar dapat menurunkan dampak dari penyebaran Covid-19 [5].

B. Pengolahan Citra Digital

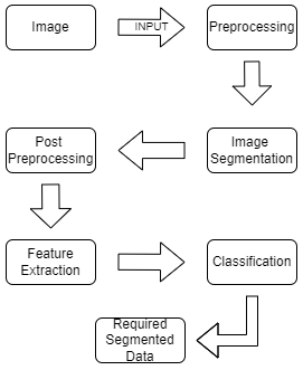
Citra adalah representasi, kemiripan, atau tiruan dari suatu objek. Gambar dapat dibagi menjadi dua jenis: gambar analog dan gambar digital. Gambar analog adalah gambar kontinu seperti gambar pada monitor TV dan sinar-X. Pengolahan citra adalah proses pengolahan piksel dalam citra digital untuk tujuan tertentu. Pengolahan citra pada awalnya dilakukan untuk meningkatkan kualitas citra, namun dalam dunia komputer ditandai dengan peningkatan daya dan kecepatan komputasi, serta munculnya ilmu komputer yang memungkinkan manusia untuk mengambil informasi dari citra [7].

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix}$$

GAMBAR 1
MATRIX PENGOLAHAN CITRA DIGITAL [8].

Secara matematis, suatu citra merupakan fungsi kontinu dengan intensitas cahaya pada bidang dua dimensi dimensi. Untuk diproses oleh komputer digital, gambar harus direpresentasikan sebagai nilai diskrit dengan angka. Mewakili fungsi kontinu dengan nilai diskrit disebut mendigitalkan gambar. Sebuah citra digital dapat direpresentasikan dengan matriks dua dimensi f (x, y) yang terdiri dari M kolom dan N baris. Perpotongan antara kolom dan baris disebut piksel (piksel = elemen gambar) atau terkecil [8].

Pemrosesan gambar digital menggunakan banyak teknik seperti koreksi, pemformatan data, penyempurnaan prosedur untuk membuat gambar dengan lebih baik kualitas. Pada dasarnya, ada empat operasi yang digunakan dalam pemrosesan gambar digital seperti sebagai gambar preprocessing, segmentasi citra, ekstraksi ciri [9].



GAMBAR 2
METODE PEMROSESAN GAMBAR [9].

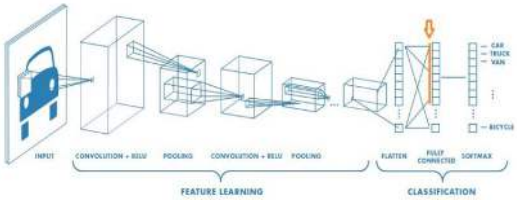
C. *Deep Learning*

Pembelajaran mendalam (pembelajaran terstruktur dalam, pembelajaran hierarkis, atau pembelajaran mesin mendalam) adalah cabang pembelajaran mesin yang didasarkan pada sekumpulan algoritme, menggunakan, atau menggunakan beberapa lapisan pemrosesan data yang ada.

Berbagai arsitektur pembelajaran mendalam seperti jaringan saraf dalam, jaringan saraf konvolusi dalam, jaringan kepercayaan mendalam, dan jaringan saraf iteratif telah diterapkan di berbagai bidang seperti visi komputer, pengenalan ucapan otomatis, pemrosesan bahasa alami, pengenalan ucapan, dan bioinformatika. Pembelajaran mendalam telah dicirikan sebagai kata kunci, atau rebranding saraf jaringan [10].

D. *Convolutional Neutral Network* (CNN)

Convolutional Neural Network (CNN) adalah salah satu bagian dari model metode deep learning, CNN awalnya dikembangkan untuk aplikasi visi komputer. CNN terdiri dari beberapa *convolutional* lapisan diikuti oleh lapisan yang terhubung penuh [11].

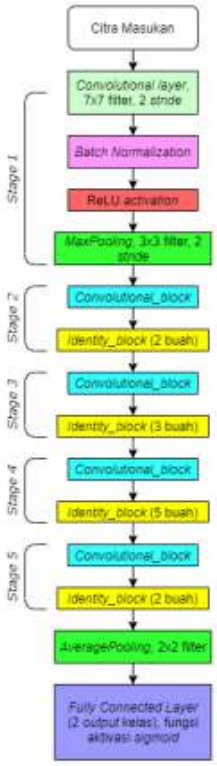


GAMBAR 3
ARSITEKTUR CNN [12].

Pada proses diatas, dapat dilihat proses CNN. Setelah citra diinput akan masuk kedalam *convolutional layer* untuk mendapatkan hasil gambar baru yang berisi features dari gambar yang di *input*, Proses *convolutional* ini akan menghasilkan feature map yang akan digunakan pada *activation layer*. Selanjutnya adalah *activation layer* tujuannya yaitu untuk meneruskan nilai yang memperlihatkan feature dari gambar inputan. Setelah itu melakukan max pool yang bertujuan untuk mengambil satu piksel pada area dengan luas tertentu pada gambar [13].

E. *Residual Network* (RESNET)

ResNet kependekan dari *Residual Network* adalah jaringan saraf klasik. Terobosan mendasar dengan ResNet adalah memungkinkan untuk melatih jaringan saraf yang sangat dalam dengan 150+ lapisan. Arsitektur Resnet menunjukkan bahwa neural network ini lebih mudah dioptimalkan, dan dapat memperoleh akurasi dari kedalaman jauh yang jauh meningkat [14].

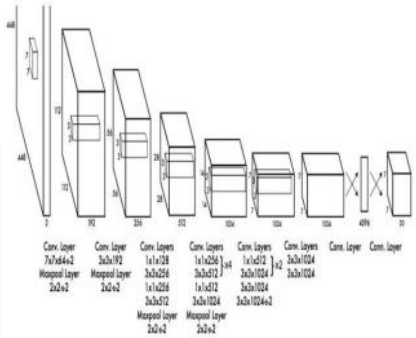


GAMBAR 4
ARSITEKTUR RESNET [14].

Pada gambar 4, ResNet memperkenalkan salah satu konsep shortcut connections dan pada konsep ini fitur yang merupakan input dari layer sebelumnya juga dijadikan sebagai input terhadap output dari layer tersebut. Dengan cara ini dilakukan sebagai salah satu solusi untuk meminimalisir hilangnya fitur-fitur penting pada saat proses konvolusi. Secara garis besar arsitektur ResNet-50 terdiri dari 5 stage proses konvolusi, untuk *convolutional_block* dan *identity_block* merupakan proses yang melakukan perulangan yang ada pada stage 1 yang mana proses ini memiliki shortcut connections didalamnya, yang kemudian dilanjutkan *average pooling* dan diakhiri dengan *fully connected layer* sebagai layer prediksi.

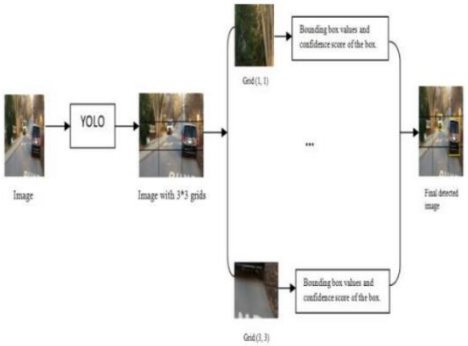
F. *You Only Look Once* (YOLO)

You Only Look Once (YOLO) adalah pendekatan baru untuk menggambar kotak yang melompat di sekitarnya sambil menampilkan beberapa objek dalam gambar secara real time. Karena melewati gambar hanya sekali melalui algoritma CNN dan mendapatkan output. R-CNN relatif mirip, tetapi YOLO secara substansial jauh lebih cepat daripada Faster R-CNN karena arsitekturnya yang sederhana. Berbeda dengan Faster R-CNN, YOLO dapat mengklasifikasikan dan melakukan regresi bounding box secara bersamaan.



GAMBAR 5
ARSITEKTUR YOLO [15].

Pada gambar 5, convolutional dilakukan untuk mengurangi dimensi spasial menjadi 7x7 dengan 1024 saluran keluaran untuk setiap lokasi. Menggunakan dua buah lapisan yang sepenuhnya telah terhubung, kemudian melakukan regresi linier ke buat prediksi kotak pembatas 7x7x2. Pada akhirnya, prediksi yang dilakukan dibuat untuk mempertimbangkan skor *confident* yang memiliki nilai tertinggi dari sebuah kotak.



GAMBAR 6
CARA KERJA YOLO [15].

Dengan YOLO, label kelas yang berisi objek, lokasinya dapat diprediksi dalam satu pandangan. Sepenuhnya menyimpang dari CNN yang khas pipa, YOLO memperlakukan deteksi objek sebagai regresi masalah dengan memisahkan kotak pembatas secara spasial dan probabilitas kelas terkait, yang diprediksi menggunakan jaringan syaraf. Proses melakukan kedua bounding ini prediksi kotak dan perhitungan probabilitas kelas adalah satu kesatuan arsitektur jaringan yang pertama kali diperkenalkan oleh YOLO [15].

G. *Euclidean Distance*

Euclidean Distance adalah jarak antara titik-titik pada garis lurus. Metode jarak ini menggunakan Teori Pitagoras. Dan merupakan perhitungan jarak yang paling sering digunakan dalam proses mesin pembelajaran. Rumus Jarak Euclidean adalah hasil dari akar kuadrat dari perbedaan dua vector [16].

Untuk pengukuran jarak antara person dapat digunakan dengan rumus berikut [17]:

$$Distance = \sqrt{(x_2 + x_1)^2 + (y_2 + y_1)^2}$$

Keterangan :
Distance = Jarak antara dua buah titik
x1 = Koordinat x untuk titik pertama
x2 = Koordinat x untuk titik kedua
y1 = Koordinat y untuk titik pertama
y2 = Koordinat y untuk titik kedua

H. *Transfer Learning*

Transfer Learning adalah sebuah metode yang menggunakan suatu model pre-trained atau bisa dikatakan telah terbentuk dan dilatih oleh suatu dataset dan kemudian model tersebut digunakan kembali untuk permasalahan yang berbeda dengan mengganti atau memodifikasi parameter atau konfigurasinya sesuai dengan dataset yang baru.

I. Konfigurasi Variabel

Variabel konfigurasi pada model yang digunakan merupakan komponen penting yang perlu disesuaikan dengan dataset dan jumlah iterasi yang diinginkan, untuk menentukan tingkat keberhasilan dari proses training dataset. Berikut adalah beberapa poin-poin penting diantaranya :

1. *Max Batches*

Max batches adalah sebuah proses untuk menentukan banyaknya iterasi yang akan dilakukan pada proses pelatihan data. Semakin tinggi nilai *max batches*, maka sistem akan semakin banyak mempelajari data latih. Jumlah training data tidak boleh lebih dari jumlah max batches. Karena jumlah data latih juga tidak perbolehkan melebihi nilai max batches. Cara menentukan *max batches* adalah sebagai berikut:

Max Batches = number of classes * 2000

2. *Subdivisions*

Subdivisions membagi nilai batch menjadi lebih kecil lagi, dan dapat disebut dengan *mini batch*. Proses ini bertujuan untuk mempercepat proses pelatihan sekaligus dengan meningkatkan akurasi menggunakan bantuan GPU. Jika menggunakan nilai *batch* sebesar 64 dan dibagi dengan 8 subdivisions, maka menghasilkan nilai 8 yang berarti dilakukan proses pelatihan untuk 8 gambar tiap bagian yang lebih kecil.

3. *Batch Size*

Batch Size merupakan variabel yang menentukan banyaknya jumlah sampel data yang akan dilatih. Semakin kecil nilai *batch size* yang digunakan, maka proses training akan makin cepat. Sementara semakin besar nilai *batch size*, proses training pun akan memakan waktu yang cukup lama karena membutuhkan kapasitas penyimpanan yang lebih banyak.

4. *Learning Rate*

Learning Rate merupakan sebuah *hyperparameter* yang menentukan seberapa banyak perubahan atau pembaharuan dari model selama proses training dilakukan. Tingkat pembelajaran juga menentukan tingkat iterasi, sehingga fungsi kerugian minimal dapat dicapai. Semakin tinggi tingkat pembelajaran, semakin cepat proses pelatihan. Dengan nilai *learning rate* yang terlalu tinggi dapat menyebabkan nilai *loss function* turun-naik tidak menentu, sehingga dibutuhkan beberapa kali percobaan untuk mendapatkan nilai *learning rate* yang optimal [18].

5. *Channels*

Nilai *channel* menentukan kedalaman citra dari data yang digunakan pada proses pelatihan. Jika data yang dilatih menggunakan citra RGB (*Red, Green, Blue*) maka variabel *channel* akan bernilai tiga. Sementara jika menggunakan citra *greyscale*, maka *channel* akan menggunakan nilai satu.

J. Parameter Performansi

Parameter performansi adalah parameter-parameter yang akan didapatkan setelah melakukan proses training. Parameter ini berdampak pada performa dari sistem yang dibuat, berikut adalah parameter-parameter yang digunakan:

1. *Confusion Matrix*

Confusion matrix adalah sebuah tabel untuk mengukur performa dari model yang dilatih. *Confusion matrix*

memiliki 4 istilah, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).

TABEL 1
CONFUSION MATRIX

		Actual Class	
		Positive	*Negative*
Predicted Class	*Positive*	TP	FP
	Negative	FN	TN

2. Akurasi

Akurasi adalah salah satu parameter yang menentukan tingkat keberhasilan model dalam mendeteksi objek *person*. Untuk menghitung akurasi, dapat menggunakan dua persamaan. Persamaan pertama dapat dilihat pada persamaan dibawah ini.

$$A = \frac{\sum B}{\sum n}$$

Dimana variabel A adalah nilai akurasi, variabel B adalah jumlah objek yang terdeteksi dengan benar, dan variabel n adalah keseluruhan data. Akurasi dapat dihitung melalui data dari *confusion matrix* dengan rumus seperti persamaan berikut:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN}\ x\ 100\%$$

3. Presisi

Presisi adalah rasio prediksi benar dan positif dari keseluruhan objek yang terdeteksi yang bernilai positif. Persamaan untuk mencari nilai presisi juga memakai data dari hasil confusion matrix dimana nilai yang diambil adalah jumlah *True Positive* dan *False Positive*. Dapat dilihat dari persamaan dibawah ini:

$$Presisi = \frac{TP}{TP + FP}\ x\ 100\%$$

4. Recall

Recall adalah rasio untuk prediksi dengan nilai benar positif dengan keseluruhan data atau objek terdeteksi dengan benar positif. Nilai recall yang tinggi menunjukkan bahwa sistem dapat mengklasifikasikan kelas objek dengan benar. Persamaan untuk mencari nilai recall dapat dilihat dibawah ini:

$$Recall = \frac{TP}{TP + FN}\ x\ 100\%$$

5. F1 score

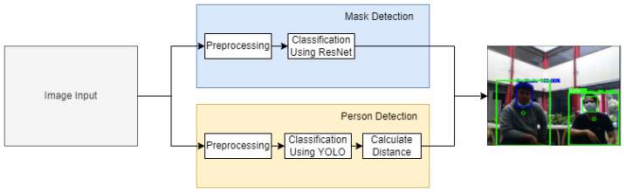
F1 *score* adalah perbandingan rata-rata dari nilai presisi dan *recall*. F1 *score* memiliki nilai tertinggi sebesar 1 dan terendah sebesar 0. Nilai F1 *score* yang semakin mendekati 1, menunjukkan bahwa kinerja sistem telah baik. Persamaan F1 *score* dapat dilihat dibawah ini:

$$F1\ Score = 2\ x\ \frac{Recall\ x\ Presisi}{Recall + Presisi}\ x\ 100\%$$

III. METODE

A. Gambaran Umum Sistem

Sistem yang akan dirancang ini adalah sistem yang akan mendeteksi pelanggaran *social distancing* dan pelanggaran penggunaan masker dengan melakukan simulasi direstoran dengan menggunakan *face detection* sebagai deteksi wajah serta model algoritma ResNet sebagai deteksi penggunaan masker dan objek *detection* sebagai deteksi person menggunakan model algoritma You Only Look Once (YOLO) serta euclidean distance sebagai deteksi *social distancing* antar objek *person*. Untuk penggunaan sistem ini menggunakan website untuk penguna memantau keadaan didalam restoran, dengan 2 fitur yang pertama yaitu start akan langsung membuka kamera dan langsung tampil pada halaman website sekaligus langsung menangkap gambar, mengkalkulasikan, dan menyimpulkan terjadi pelanggaran social distancing dan penggunaan masker atau tidak dan fitur yang kedua yaitu stop akan langsung menutup kamera yang terhubung langsung pada *website*.



GAMBAR 7
DESAIN SISTEM.

B. Deteksi *Social Distancing*

Pada pendeteksian *social distancing* ini akan menggunakan data latih yang menggunakan model yang digunakan sebelumnya dan sudah dilatih sehingga dapat mendeteksi lebih dari 80 kelas objek yang berbeda. Oleh karena itu diperlukan adanya *transfer learning* untuk menyesuaikan model yang dibuat agar dapat dipakai menjadi *pretrained* model, dan mampu mendeteksi hanya 1 kelas saja yaitu kelas *person*. Dataset yang digunakan untuk melatih model sehingga hanya dapat mendeteksi *person* diambil dari *open image* dataset dengan jumlah dataset yang diambil adalah sebanyak 1800 gambar. Berikut tabel dataset yang dipakai dan contoh dari dataset untuk kelas *person*:

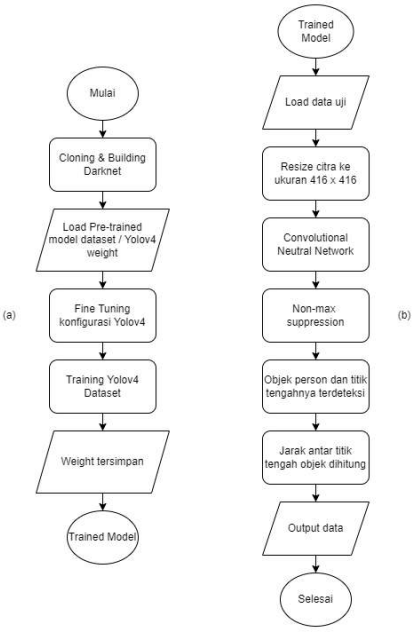
GAMBAR 8
CONTOH DATA LATIH *PERSON*

TABEL 2
DATASET YOLO

Data Latih	Data Validasi	*Classes*
1620	180	*Person*
1440	360	*Person*
1260	540	*Person*

Pada tabel 2 dapat dilihat untuk dataset YOLO pembagian data latih dan data validasi atau bisa disebut data test. Dengan menspesifikasikan satu kelas saja, yaitu kelas *person*.

Berikut adalah alur proses dari proses pelatihan pada model YOLO dapat dilihat pada gambar dibawah ini.



GAMBAR 9
FLOWCHART TRAINING DAN *TEST* DARI SISTEM YOLOV4

Pada *flowchart* gambar 9, pada bagian (a) proses pelatihan yang dilakukan terhadap dataset. Proses dimulai dengan membangun Darknet sebagai arsitektur Yolov4, kemudian akan dilakukan penyesuaian dan fine-tuning pada *configuration file* dari *pre-trainined* Yolov4 model sebagai data latih baru, setelah itu pelatihan dapat dilakukan. Keluaran yang dihasilkan akan menjadi sebuah *weight file* yang akan tersimpan dan digunakan pada sistem.

Pada bagian (b) memperlihatkan alur sistem mendeteksi objek sehingga dapat menghitung jarak antar objek untuk menentukan pelanggaran *social distancing*. Keluaran hasil pelatihan akan mempengaruhi akurasi saat dilakukan testing. Sistem akan membaca *weight file* yang merupakan model Yolov4 yang telah dilatih untuk melakukan deteksi objek beserta perhitungan jarak antar objek, dan mendapatkan hasil deteksi.

C. Deteksi Penggunaan Masker

Pada pendeteksian penggunaan masker ini akan menggunakan data latih yang menggunakan model yang digunakan sebelumnya dan sudah dilatih sehingga dapat mendeteksi dengan akurat. Oleh karena itu diperlukan adanya *transfer learning* untuk menyesuaikan model yang dibuat agar dapat dipakai menjadi *pretrained* model. Dataset yang digunakan untuk melatih model sehingga hanya dapat mendeteksi person diambil dari *Github* dengan jumlah dataset yang diambil adalah sebanyak ±4100 gambar. Berikut tabel dataset yang dipakai dan contoh dari dataset untuk kelas *mask* dan *nonmask*:

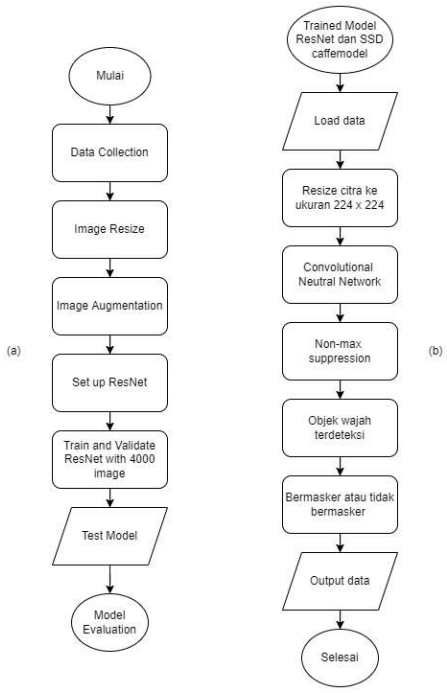
GAMBAR 10
CONTOH DATA LATIH *MASK* DAN *NONMASK*

TABEL 3
DATASET RESNET

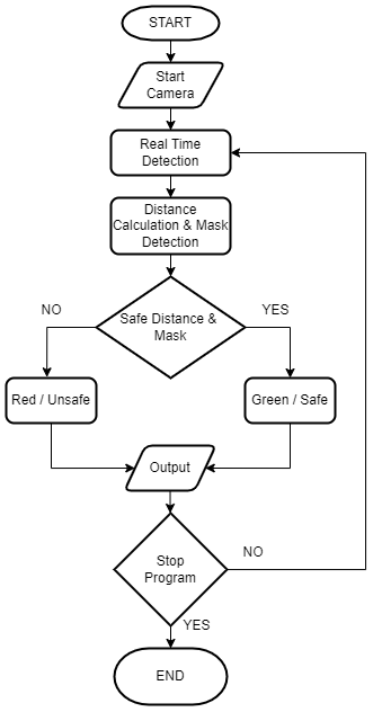
Data Latih	Data Validasi	*Classes*
3690	410	*Mask/Nonmask*
3280	820	*Mask/Nonmask*
2870	1230	*Mask/Nonmask*

Pada tabel 3.2 dapat dilihat untuk dataset ResNet pembagian data latih dan data validasi atau bisa disebut data test. Dengan menspesifikasikan dua kelas, yaitu kelas *mask* dan *nonmask*.

Berikut adalah alur proses dari proses pelatihan pada model resnet dapat dilihat pada gambar dibawah ini.



GAMBAR 11
FLOWCHART TRAINING DAN *TEST* DARI SISTEM RESNET



GAMBAR 12
FLOWCHART SISTEM KERJA

Pada flowchart diatas, pada bagian (a) proses pelatihan yang dilakukan terhadap dataset. Proses dimulai dengan mengumpulkan data, kemudian akan dilakukan penyesuaian ukuran gambar, setelah data akan diubah untuk membuat beberapa data lagi untuk proses pelatihan. Selanjutnya sesudah itu akan melakukan persiapan dan melakukan proses *training*. Keluaran yang dihasilkan akan menjadi sebuah model *file* yang akan tersimpan dan digunakan pada sistem.

Pada bagian (b) memperlihatkan alur sistem mendeteksi objek sehingga dapat menghitung jarak antar objek untuk menentukan pelanggaran penggunaan masker. Keluaran hasil pelatihan akan mempengaruhi akurasi saat dilakukan *testing*. Sistem akan membaca model file yang merupakan model ResNet yang telah dilatih untuk melakukan deteksi wajah bermasker atau tidak, dan mendapatkan hasil deteksi.

D. Alur Keseluruhan Sistem

Untuk alur kerja sistem pendeteksian pelanggaran social distancing dan penggunaan masker secara keseluruhan dapat dilihat pada diagram dibawah ini:

Langkah – langkah yang dilakukan untuk pendeteksian *social distancing* dan deteksi penggunaan masker adalah sebagai berikut:

1. Dimulai dengan menekan tombol *start* pada sistem, maka sistem akan memulai menyalakan kamera pada laptop dan menangkap gambar secara *real time*.
2. Selanjutnya sistem akan mulai melakukan proses pendeteksian *person detection* dan masker *detection* untuk memastikan kedua objek tersebut.
3. Pada proses ini sistem akan melakukan proses *distance measurement* untuk menentukan jarak aman kedua objek, sedangkan untuk deteksi penggunaan masker untuk menentukan objek tersebut sudah menggunakan masker dengan baik atau tidak.
4. Dalam proses penentuan *safe* atau *unsafe* ini, dibagi menjadi 2 kategori yaitu yang pertama *YES* (*safe*) yang berarti jarak *social distancing* dan penggunaan masker sudah baik akan masuk kedalam kategori *YES* (*safe*). Yang kedua *NO* (*unsafe*) yang berarti jarak *social distancing* dan penggunaan masker tidak baik akan masuk kedalam kategori *NO* (*unsafe*).
5. Pada proses *output* ini akan menampilkan hasil dari proses penentuan *safe* atau *unsafe* untuk jarak *social distancing* sedangkan untuk penggunaan masker akan *non mask* atau *mask*. Jika sudah memenuhi keriteria yang diberikan maka objek yang terdeteksi akan masuk dalam kategori aman, *output* dari sistem akan diberikan tulisan "*safe*" serta *frame* objek menjadi hijau. Tetapi Jika belum memenuhi keriteria yang diberikan maka objek yang terdeteksi akan masuk dalam kategori tidak

aman, *output* dari sistem akan diberikan tulisan "*unsafe*" serta *frame* objek menjadi merah.

6. Setelah proses *output* sudah keluar, *user* dapat memilih untuk memberhentikan program atau tetap menjalankan program. Jika *user* memilih memberhentikan program maka otomatis kamera akan langsung menutup.

E. Deteksi Pelanggaran *Social Distancing* dan Penggunaan Masker

Untuk deteksi pelanggaran social distancing ketika sistem sudah mendeteksi adanya objek manusia dan memberikan *bounding box* beserta titik *center* kemudian akan dilanjutkan untuk menghitung jarak antar objek dengan menggunakan acuan pada titik *center* tersebut. Contoh objek *person* yang terdapat titik *center* dapat dilihat pada gambar 13 dimana titik *center* berwarna bulat hijau yang berada didalam *bounding box*.

GAMBAR 13
OBJEK *PERSON* YANG TERDETEKSI DAN MEDAPATKAN TITIK *CENTER*

Setelah mendapatkan titik *center* maka akan dihitung dengan persamaan rumus *Euclidean Distance*, nilai yang akan dipakai untuk perhitungan ini yaitu dari pemisalan pada gambar 13 dengan nilai permisalan:

Panjang (y) Objek 1 : 950
Lebar (x) Objek 1 : 400
Panjang (y) Objek 2 : 730
Lebar (x) Objek 2 : 330

Karena dihitung dari titik center, maka nilai x dan y yang diketahui akan dibagi menjadi 2, sehingga menjadi:

y1: 475
x1: 200
y2: 365
x2: 165

maka didapat, d (x,y) = $\sqrt{(200-165)^2+(475-365)^2}$
= 115.43

Maka berdasarkan hasil permisalan tersebut, jarak antara kedua objek pada gambar 13 tersebut mendapatkan hasil 115.43 yang mana jarak tersebut tidak melanggar social distancing dan bounding box akan berwarna hijau.

Sedangkan untuk deteksi pelanggaran penggunaan masker sistem akan mencari wajah untuk dideteksi setelah mendapatkan wajah dengan nilai confident yang sesuai dengan sistem maka sistem akan langsung memberikan bounding box pada bagian wajah setelah menentukan apakah wajar tersebut bermasker atau tidak bermasker. Contoh wajah yang telah terdeteksi dan diberikan bounding box dapat dilihat pada gambar 3.20.

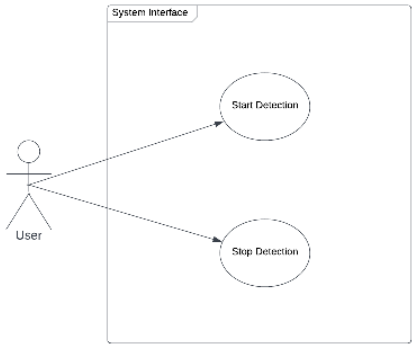
GAMBAR 14
WAJAH YANG BERHASIL DIDETEKSI DAN MENDAPATKAN BOUNDING BOX

F. Perancangan Web

Web berfungsi sebagai interface yang berguna untuk mempermudah penggunaan sistem tanpa harus melakukan run program berulang-ulang. Untuk mengetahui secara ringkas bagaimana penggunaan web ini, dapat dilihat pada gambar *use case* dan *activity* diagram berikut:

1. *Use case* diagram

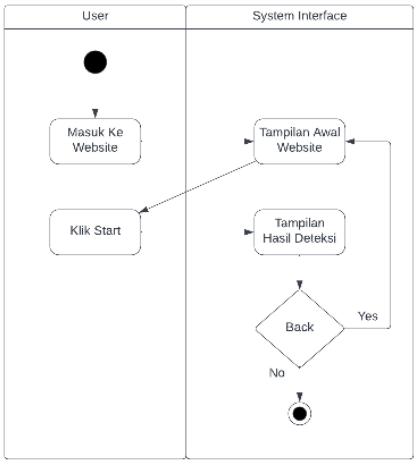
Use Case bertujuan untuk menjelaskan interaksi diantara user dengan sistem yang dibuat, serta digunakan untuk mengetahui fungsi yang terdapat pada sistem. Berikut adalah gambar use case pada gambar 15.



GAMBAR 15
USE CASE DIAGRAM

2. *Activity* diagram

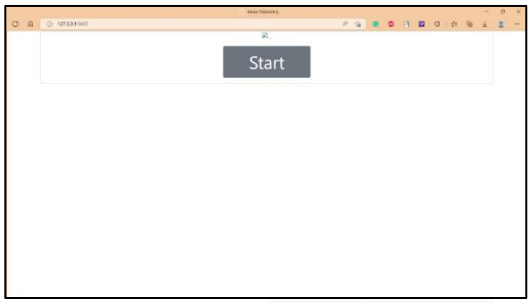
Diagram yang menjelaskan alur kerja sistem yang dijalankan, berikut gambar diagram *activity* pada gambar 16.



GAMBAR 16
ACTIVITY DIAGRAM

3. Tampilan website

Berikut akan menampilan interface yang digunakan untuk menampilan *output* video yang telah diproses oleh sistem, website ini menggunakan *framework* "flask".



GAMBAR 17
TAMPILAN AWAL WEBSITE

IV. HASIL DAN PEMBAHASAN

A. *Training Model* Yolo

Berdasarkan dataset yang digunakan untuk melakukan proses *training*. Untuk dataset akan dibagi menjadi tiga rasio pengujian, yaitu pengujian rasio 90%:10%, rasio 80%:20%, rasio 70%:30%. Kemudian proses *training* akan dilakukan di Google Colab Notebooks, yang mana sudah berbasis *cloud* untuk pemrograman python. Sebelum proses training dimulai diperlukan untuk konfigurasi sistem sebelum melatih model baru. Berikut adalah konfigurasi sistem yang diubah:

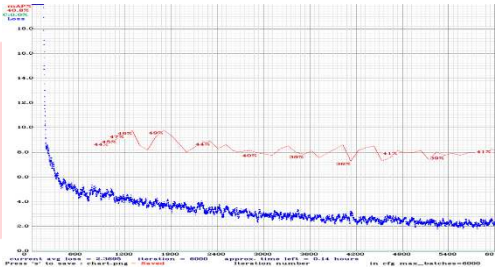
Batch Size = 64
Subdivisions = 16
Learning Rate = 0.001
Max Batches = 6000
Classes = 1

1. Pengujian Beradasarkan Partisi Data

Pengujian berdasarkan partisi data dilakukan untuk mengetahui kinerja maksimum yang dilakukan oleh sistem klasifikasi yang telah dibuat. Prosesnya dilakukan dengan cara membagi data ke dalam data latih dan data uji. Perbandingan dari kedua data dapat dilihat pada tabel di bawah ini.

TABEL 4
PERBEDAAN PARTISI DATA UNTUK PENGUJIAN YOLO

Percobaan Ke-	Data Latih	Data Uji	Durasi Latih
1	90%	10%	± 12 jam
2	80%	20%	± 11 jam
3	70%	30%	± 10 jam



GAMBAR 19
GRAFIK MAP 1 RASIO 90%:10%

Pada gambar 19 berdasarkan hasil grafik pada rasio 90%:10% tidak mendapatkan kendala dalam proses *training* menjadikan gambar grafik diatas tampil secara penuh dan tidak putus-putus. Mendapatkan hasil mAP dan grafik yang bagus.

TABEL 5
HASIL TRAINING RASIO BERBEDA

Rasio	Presisi	Recall	F1 Score	Average IoU	mAP
90% : 10%	42%	61%	49%	32.59%	49.02%
80% : 20%	32%	66%	43%	23.58%	41.65%
70% : 30%	48%	56%	52%	37.55%	44.51%

Pada tabel 5 hasil kesimpulan dari *training* rasio berbeda, mendapatkan hasil yang paling baik yaitu dengan rasio 90%:10%.

B. *Training Model Resnet*

Berdasarkan dataset yang digunakan untuk melakukan proses *training*. Untuk dataset akan dibagi menjadi tiga rasio pengujian, yaitu pengujian rasio 90%:10%, rasio 80%:20%, rasio 70%:30%. Kemudian proses *training* akan dilakukan di *Visual Studio Code* untuk pemrograman python. Sebelum proses *training* dimulai diperlukan untuk konfigurasi sistem sebelum melatih model baru. Berikut adalah konfigurasi sistem bawaan:

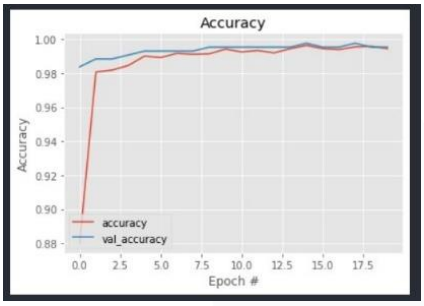
Split Data = 0.2
Learning Rate = 0.0005
Batch Size = 256

1. Pengujian Berdasarkan Partisi Data

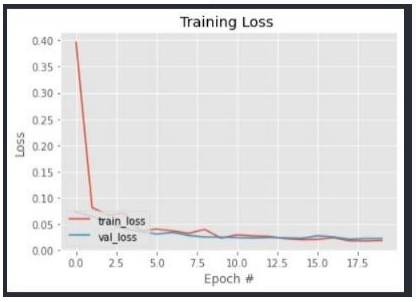
Pengujian berdasarkan partisi data dilakukan untuk mengetahui kinerja maksimum yang dilakukan oleh sistem klasifikasi yang telah dibuat. Prosesnya dilakukan dengan cara membagi data ke dalam data latih dan data uji. Perbandingan dari kedua data dapat dilihat pada tabel di bawah ini.

TABEL 6
PERBEDAAN PARTISI DATA UNTUK PENGUJIAN RESNET

Percobaan Ke-	Data Latih	Data Uji	Durasi Latih
1	90%	10%	± 1 jam
2	80%	20%	± 50 menit
3	70%	30%	± 40 menit



GAMBAR 20
GRAFIK AKURASI 1 RASIO 90%:10%



GAMBAR 21
GRAFIK TRAIN LOSS 1 RASIO 90%:10%

Pada gambar 20 dan gambar 21 berdasarkan grafik akurasi dan grafik train loss untuk rasio 90%:10%, mendapatkan hasil terbaik dan mendapatkan grafik yang baik karena tidak terjadi kenaikan dan penurunan yang terlalu jauh pada grafik.

TABEL 7
HASIL TRAINING RASIO BERBEDA

Rasio	Akurasi	Presisi	Recall	F1 Score
90% : 10%	99.34	99.04	99.15	99.09
80% : 20%	99.44	99.34	99.34	99.34
70% : 30%	99.53	99.28	99.31	99.30

Pada tabel 7 dapat dilihat untuk hasil training rasio berbeda ResNet disetiap percobaan masing-masing rasio mendapatkan hasil yang rata-rata diatas 99%.

C. Pengujian *Social Distancing*

Setelah melakukan beberapa kali *training* untuk mencari akurasi yang terbaik didapatkan hasil *weight file*, kemudian *weight file* tersebut yang akan digunakan sebagai model pendeteksian. Kemudian program akan dijalankan untuk melihat performa model dalam mendeteksi objek yaitu *person*.

GAMBAR 22
CONTOH PENGUJIAN SUDUT 0° SKENARIO SEJAJAR

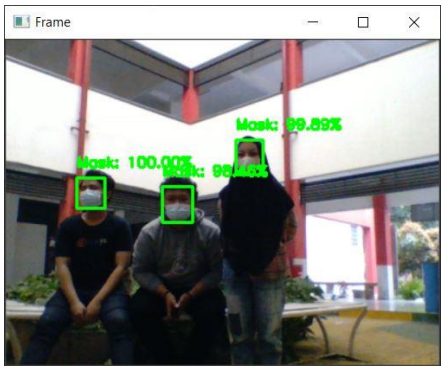
Berikut tampilan hasil-hasil pengujian yang dilakukan sesuai dengan tabel dibawah ini.

TABEL 8
HASIL PENGUJIAN SOCIAL DISTANCING

No	Sudut	Jarak(cm)	Diterima	Tidak
1	0°	100	✓	
2	0°	150	✓	
3	0°	200	✓	
4	30°	100	✓	
5	30°	150	✓	
6	30°	200	✓	
7	60°	100	✓	
8	60°	150	✓	
9	60°	200	✓	

D. Pengujian Penggunaan Masker

Setelah melakukan beberapa kali *training* untuk mencari akurasi yang terbaik didapatkan hasil model *file*, kemudian model *file* tersebut yang akan digunakan sebagai model pendeteksian. Kemudian program akan dijalankan untuk melihat performa model dalam mendeteksi wajah yaitu *mask and nonmask*.



GAMBAR *23*
CONTOH PENGUJIAN JARAK 200CM SKENARIO SEJAJAR

Berikut tampilan hasil-hasil pengujian yang dilakukan sesuai dengan tabel dibawah ini.

TABEL 11
HASIL PENGUJIAN MASKER

No	Sudut	Jarak(cm)	Diterima	Tidak
1	0°	100	✓	
2	0°	150	✓	
3	0°	200	✓	
4	30°	100	✓	
5	30°	150	✓	
6	30°	200	✓	
7	0°	100		✓
8	0°	150	✓	
9	0°	200	✓	

V. KESIMPULAN

Sistem untuk pendeteksian objek person dapat dilakukan dengan menggunakan algoritma YOLO dengan baik dan mendapatkan mAP sebesar 49.04%, presisi 42%, dan recall 61%. Menggunakan rasio partisi data 90%:10% dengan *max batches* 6000 dengan *learning rate* 0.001.

Sistem untuk pendeteksian objek *mask/nonmask* dapat dilakukan dengan menggunakan algoritma ResNet dengan baik dan mendapatkan Akurasi sebesar 99.34%, presisi 99.04%, dan *recall* 99.15%. Menggunakan rasio partisi data 90%:10% dengan *learning rate* 0.0001, dan *batch size* 256.

Telah berhasil melakukan beberapa kali pengujian *hyper parameter* untuk algoritma YOLO hasil terbaik didapatkan dengan *max batches* 1000 dan learning rate 0.002 mendapatkan mAP sebesar 46.99%. Sedangkan untuk algoritma ResNet hasil terbaik didapatkan dengan *learning rate* 0.0003 dan batch size 256 mendapatkan akurasi sebesar 99.34%.

REFERENSI

[1] T. Singhal, "A Review of Coronavirus Disease-2019 (COVID-19)," *Indian Journal of Pediatrics*, vol. 87, no. 4. Springer, pp. 281–286, Apr. 01, 2020. doi: 10.1007/s12098-020-03263-6.

[2] A. Saehana and A. Kadri, "An Analysis of Compliance Level of Health Protocol Implementation at the Population Control and Family Agency in Central Sulawesi During the Covid-19 Pandemic," *International Journal of Health, Economics, and Social Sciences*, vol. 3, 2021.

[3] N. S. Patil, K. Rani, S. Rangappa, and V. Jain, "Social Distancing Detection," 2021. [Online]. Available: www.ijres.org

[4] A. Negi, K. Kumar, P. Chauhan, and R. S. Rajput, "Deep Neural Architecture For Face Mask Detection On Simulated Masked Face Dataset Against Covid-19 Pandemic," in *Proceedings - IEEE 2021 International Conference on Computing, Communication, and Intelligent Systems, ICCCIS 2021*, Feb. 2021, pp. 595–600. doi: 10.1109/ICCCIS51004.2021.9397196.

[5] Rucha Visal, Atharva Theurkar, and Bhairavi Shukla, *Monitoring Social Distancing for Covid-19 Using OpenCV and Deep Learning*, vol. 7. International Research Journal of Engineering and Technology (IRJET), 2020.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Dec. 2016, vol. 2016-December, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[7] R. D. Kusumanto and A. N. Tompunu, "PENGOLAHAN CITRA DIGITAL UNTUK MENDETEKSI OBYEK MENGGUNAKAN PENGOLAHAN WARNA MODEL NORMALISASI RGB," 2011.

[8] P. Paygude, R. Garg, P. Pathak, A. Trivedi, and A. Raj, "IMAGE PROCESSING USING MACHINE LEARNING," *IJSDR2009078 International Journal of Scientific Development and Research*, 2020, [Online]. Available: www.ijsdr.org

[9] A. Professor and R. B. Banu, "Digital Image Processing Techniques-A Review," 2019. [Online]. Available: www.jetir.org

[10] B. B. Benuwa, Y. Zhan, B. Ghansah, D. K. Wornyo, and F. B. Kataka, "A Review of Deep Machine Learning," *International Journal of Engineering Research in Africa*, vol. 24. Trans Tech Publications Ltd, pp. 124–136, 2016. doi: 10.4028/www.scientific.net/JERA.24.124.

[11] Dileep P, Dibyajyoti Das, and Prabin Kumar Bora, *Dense Layer Dropout Based CNN Architecture for Automatic Modulation Classification*. IEEE, 2020.

[12] Mrs. Arpana Mahajan and Dr. Sanjay Chaudhary, *Categorical Image Classification Based On*

Representational Deep Network (RESNET). Proceedings of the Third International Conference on Electronics, Communication and Aerospace Technology (ICECA 2019) : 12-14, June 2019, 2019.

[13] Alif Bin Abdul Qayyum, Tanveerul Islam, and Md. Aynal Haque, *Malaria Diagnosis with Dilated Convolutional Neural Network Based Image Analysis*. 2019.

[14] Faiz Nashrullah, Suryo Adhi Wibowo, and Gelar Budiman, "The Investigation of Epoch Parameters in ResNet-50 Architecture for Pornographic Classification," *Journal of Computer, Electronic, and Telecommunication*, vol. 1, no. 1, Jul. 2020, doi: 10.52435/complete.v1i1.51.

[15] N. Geethapriya. S, Duraimurugan, and S.P. Chokkalingam, "Real-Time Object Detection with Yolo," 2019.

[16] R. Suwanda, Z. Syahputra, and E. M. Zamzami, "Analysis of Euclidean Distance and Manhattan Distance in the K-Means Algorithm for Variations Number of Centroid K," in *Journal of Physics: Conference Series*, Jul. 2020, vol. 1566, no. 1. doi: 10.1088/1742-6596/1566/1/012058.

[17] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean Distance Matrices: Essential theory, algorithms, and applications," *IEEE Signal Process Mag*, vol. 32, no. 6, pp. 12–30, Nov. 2015, doi: 10.1109/MSP.2015.2398954.

[18] J. Konar, P. Khandelwal, and R. Tripathi, "Comparison of Various Learning Rate Scheduling Techniques on Convolutional Neural Network," Feb. 2020. doi: 10.1109/SCEECS48394.2020.94.