

SKRIPSI

**IMPLEMENTASI ALGORITMA *OBJECT DETECTION*
YOLOV4 DAN *EUCLIDEAN DISTANCE* DALAM
MENDETEKSI PELANGGARAN *SOCIAL DISTANCING***

Disusun dan diajukan oleh

Muhammad Amirullah Zulkiflie

H071171526



**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN
ALAM
UNIVERSITAS HASANUDDIN
2021**

**IMPLEMENTASI ALGORITMA OBJECT
DETECTION YOLOV4 DAN EUCLIDEAN DISTANCE
DALAM MENDETEKSI PELANGGARAN SOCIAL
DISTANCING**

SKRIPSI

**Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana
Komputer pada Program Studi Sistem Informasi Departemen Matematika
Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin**

MUHAMMAD AMIRULLAH ZULKIFLIE

H071171526

PROGRAM STUDI SISTEM INFORMASI DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2021

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Muhammad Amirullah Zulkiflie

NIM : H071171526

Program Studi : Sistem Informasi

Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

**Implementasi Algoritma Object Detection YOLOv4 dan Euclidean Distance
dalam Mendeteksi Pelanggaran Social Distancing**

adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain dan belum pernah dipublikasikan dalam bentuk apapun.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 15 Oktober 2021

Yang menyatakan,



Muhammad Amirullah Zulkiflie

NIM. H071171304

**IMPLEMENTASI ALGORITMA OBJECT DETECTION
YOLOV4 DAN EUCLIDEAN DISTANCE DALAM
MENDETEKSI PELANGGARAN SOCIAL DISTANCING**

Disusun dan diajukan oleh:

MUHAMMAD AMIRULLAH ZULKIFLIE

H071171526

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin pada tanggal 15 Oktober 2021 dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama

Pembimbing Pertama

Dr. Hendra, S.Si., M.Kom.

NIP. 19760102 200212 1 001

Supri Bin Hj Amir, S.Si., M.Eng.

NIP. 19880504 201903 1 012

Ketua Program Studi,

Dr. Muhammad Hasbi, M.Sc

NIP.196307201989031003



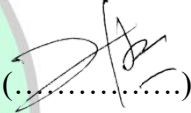

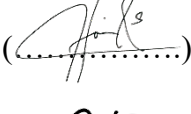

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama : Muhammad Amirullah Zulkiflie
NIM : H071171526
Program Studi : Sistem Informasi
Judul Skripsi : Implementasi Algoritma Object Detection YOLOv4 dan Euclidean Distance dalam Mendeteksi Pelanggaran Social Distancing

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

DEWAN PENGUJI

| | | Tanda tangan |
|------------|-------------------------------------|---|
| Ketua | : Dr. Hendra, S.Si., M.Kom. |  |
| Sekretaris | : Supri Bin Hj. Amir, S.Si., M.Eng. |  |
| Anggota | : Dr. Muhammad Hasbi, M.Sc. |  |
| Anggota | : A. Muh. Amil Siddik, S.Si., M.Si. |  |

Ditetapkan di : Makassar

Tanggal : 15 Oktober 2021



KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah *Subbhanahu Wa Ta'ala* karena berkat rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir pendidikan jenjang Strata 1 pada Program Studi Sistem Informasi, Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin. Dengan judul tugas akhir “Implementasi Algoritma *Object Detection YOLOv4* dan *Euclidean Distance* dalam Mendeteksi Pelanggaran *Social Distancing*”.

Penulis menyadari bahwa dalam penyusunan dan penulisan laporan tugas akhir mendapatkan banyak bantuan serta bimbingan dari berbagai pihak, dari masa perkuliahan hingga pada tahap akhir. Oleh karena itu, pada kesempatan ini, dengan segala kerendahan hati penulis menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Rektor Universitas Hasanuddin, Ibu **Prof. Dr. Dwia Aries Tina Pulubuhu, M.A.** beserta jajarannya; Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, Bapak **Dr. Eng. Amiruddin, S.Si.** beserta jajarannya; Ketua Departemen Matematika, Bapak **Prof. Dr. Nurdin, S.Si., M.Si.** beserta jajarannya; Ketua Program Studi Sistem Informasi, Bapak **Dr. Muhammad Hasbi, M.Sc.** beserta jajarannya; serta Bapak/Ibu dosen Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin atas ilmu-ilmu dan bantuannya dalam berbagai bentuk.
2. Kedua Orang Tua penulis, Ayahanda **Zulkiflie Marauni SH.**, Ibunda **Zulfiani** dan keluarga lainnya yang selalu menjadi motivasi penulis hingga dapat berada pada tahap ini dan dapat menyelesaikan penyusunan tugas akhir.
3. Bapak **Dr. Hendra, S.Si., M.Kom.** sebagai pembimbing utama dan Bapak **Supri Bin Hj Amir, S.Si., M.Eng.** sebagai pembimbing pertama sekaligus pendamping akademik yang senantiasa menyediakan waktu, pikiran; dan tenaga, dalam pengembangan tugas akhir ini.
4. Bapak **Dr. Muhammad Hasbi, M.Sc** Selaku Penguji I dan Bapak **A. Muh. Amil Siddik, S.Si., M.Si.** selaku Penguji II atas waktu dan kesediaannya

sebagai penguji untuk tugas akhir ini. Terima kasih pula untuk ilmu-ilmu yang telah diberikan selama proses perkuliahan.

5. **Rafly Ahmad Mubin** dan **Muh. Ramadhani Pratama S. Limpo** yang telah Membantu dalam sebagai model pada penelitian ini.
6. Saudara-saudara sesama mahasiswa **Sistem Informasi** khususnya **Angkatan 2017** atas kebersamaan dan segala suka dukanya selama menjadi mahasiswa Universitas Hasanuddin.
7. Seluruh pihak yang tidak dapat disebutkan satu per satu atas segala bentuk kontribusi, partisipasi, serta motivasi yang diberikan kepada penulis selama ini. Akhir kata, penulis memohon maaf apabila terdapat kesalahan dalam penyusunan tugas akhir ini dan semoga membawa manfaat bagi pembaca.

Makassar, 15 Oktober 2021

Yang menyatakan,



Muhammad Amirullah Zulkiflie

NIM. H071171526

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama : Muhammad Amirullah Zulkiflie
NIM : H071171526
Program Studi : Sistem Informasi
Departemen : Matematika
Fakultas : Matematika dan Ilmu Pengetahuan Alam
Jenis karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul:

Implementasi Algoritma *Object Detection YOLOv4* Dan *Euclidean Distance* Dalam Mendeteksi Pelanggaran *Social Distancing*

beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar pada tanggal 15 Oktober 2021

Yang menyatakan


(Muhammad Amirullah Zulkiflie)

ABSTRAK

Pandemi COVID-19 (*Coronavirus Disease-19*) dianggap sebagai ancaman kesehatan masyarakat terbesar saat ini, sehingga publik diwajibkan untuk mematuhi *social distancing*. Menurut WHO, saat melakukan aktivitas di publik orang-orang diwajibkan untuk menjaga jarak sekitar 6 kaki atau 1,8 meter. Penelitian ini berfokus dalam membuat sebuah sistem yang dapat melakukan deteksi pelanggaran *social distancing* menggunakan algoritma object detection *YOLOv4*. Dari hasil pengujian model, Model *YOLOv4* berhasil dalam mendeteksi objek manusia, dan mendapatkan skor mAP dengan nilai sebesar 78,6 %, dalam pengujian deteksi manusia menggunakan video digital model berhasil mendeteksi manusia dengan hasil kecepatan deteksi terbaik 51,2 FPS . Dalam mendeteksi pelanggaran *social distancing* model dapat melakukan deteksi pelanggaran *social distancing* dengan baik dengan metode konversi pixel rata-rata error jarak antara objek yang didapatkan adalah 0,064 meter.

Kata kunci: *YOLO*, Deteksi Objek, *Social Distancing*, *CNN*, *Deep Learning*.

ABSTRACT

The COVID-19 (Coronavirus Disease-19) pandemic is considered the biggest public health threat today, so the public is required to adhere to social distancing. According to WHO, when carrying out activities in public, people are required to maintain a distance of about 6 feet or 1.8 meters. This study focuses on creating a system that can detect social distancing violations using the YOLOv4 object detection algorithm. From the results of model testing, the YOLOv4 model was successful in detecting human objects, and got an mAP score with a value of 78.6%, in testing human detection using digital video the model succeeded in detecting humans with the best detection speed of 51.2 FPS. In detecting social distancing violations, the model can detect social distancing violations with the pixel conversion method, the average error distance between objects obtained is 0.064 meters.

Keywords: YOLO, Object Detection, Social Distancing, CNN, Deep Learning.

DAFTAR ISI

| | |
|--|------|
| PERNYATAAN KEASLIAN | II |
| HALAMAN PENGESAHAN | IV |
| KATA PENGANTAR | V |
| PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR..... | VII |
| ABSTRAK | VIII |
| ABSTRACT | IX |
| DAFTAR ISI | X |
| DAFTAR GAMBAR | XII |
| DAFTAR TABEL | XIII |
| BAB I PENDAHULUAN | 1 |
| 1.1. Latar Belakang..... | 1 |
| 1.1. Rumusan Masalah..... | 4 |
| 1.2. Tujuan Penelitian..... | 4 |
| 1.3. Manfaat Penelitian..... | 4 |
| 1.4. Batasan Masalah | 5 |
| BAB II TINJAUAN PUSTAKA | 6 |
| 2.1. Social Distancing..... | 6 |
| 2.2. Citra Digital..... | 6 |
| 2.3. Video Digital | 8 |
| 2.4. Deep Learning | 8 |
| 2.5. Convolutional Neural Network (CNN)..... | 8 |
| 2.6.1. Convolutional Layer | 10 |
| 2.6.2. Fungsi Aktivasi..... | 11 |
| 2.6.3. Pooling Layer | 13 |
| 2.6.4. Fully Connected Layer..... | 14 |
| 2.6. You Only Look Once (YOLO)..... | 15 |
| 2.7. Metrik Perhitungan Performa Model Object Detection..... | 19 |
| 2.10.1. Mean Average Precision (mAP)..... | 20 |
| 2.10.2. Frame Per Second | 22 |

| | |
|--|-----------|
| 2.8. Euclidean Distance | 23 |
| 2.9. Transformasi Pespektif | 23 |
| 2.10. Penelitian Terdahulu..... | 24 |
| BAB III METODOLOGI PENELITIAN..... | 28 |
| 3.1. Waktu dan Lokasi Penelitian..... | 28 |
| 3.2. Instrumen Penelitian | 28 |
| 3.3. Tahapan Penelitian..... | 29 |
| 3.4. Alur Proses Pengembangan Sistem Deteksi Social Distancing | 31 |
| BAB IV HASIL DAN PEMBAHASAN | 33 |
| 4.1. Deskripsi Data | 33 |
| 4.2. Preprocessing Data | 33 |
| 4.3. Perancangan Model dan Pelatihan Model..... | 35 |
| 4.3.1. Persiapan Lingkungan Pelatihan Cloud..... | 35 |
| 4.3.2. Konfigurasi Model..... | 36 |
| 4.3.1. Pelatihan Model..... | 38 |
| 4.4. Analisis Model dan Pengujian Model..... | 38 |
| 4.5. Perancangan Sistem Deteksi Social Distancing | 41 |
| 4.5.1. Deteksi Manusia | 42 |
| 4.5.2. Transformasi Perspektif..... | 43 |
| 4.5.3. Kalkulasi Jarak antara Objek..... | 43 |
| 4.6. Pengujian dan Analisis Hasil Sistem Deteksi Social Distancing | 45 |
| 4.6.1. Konversi Piksel..... | 48 |
| 4.6.2. Deteksi Social Distancing dengan Konversi Piksel..... | 50 |
| BAB V PENUTUP | 52 |
| 5.1. Kesimpulan..... | 52 |
| 5.2. Saran | 52 |
| Daftar Pustaka | 54 |
| LAMPIRAN | 58 |

DAFTAR GAMBAR

| | |
|---|----|
| Gambar 1.1. Pentingnya <i>Social Distancing</i> (Armstrong, 2021)..... | 2 |
| Gambar 2.1. Praktik <i>Social Distancing</i> | 6 |
| Gambar 2.2. Contoh Citra Berwarna | 7 |
| Gambar 2.3. Proses <i>Convolutional Neural Network</i> | 9 |
| Gambar 2.4. Ilustrasi Proses Convolutional Layer..... | 10 |
| Gambar 2.5. Fungsi Aktivasi Linear..... | 11 |
| Gambar 2.6. Fungsi Aktivasi <i>Leaky ReLU</i> | 12 |
| Gambar 2.7. Fungsi Aktivasi <i>Mish</i> | 13 |
| Gambar 2.8. <i>Pooling Layer</i> | 14 |
| Gambar 2.9. Ilustrasi <i>Fully Connected Layer</i> | 15 |
| Gambar 2.10. Ilustrasi Cara Kerja YOLO (Redmon dkk, 2016)..... | 16 |
| Gambar 2.11. Arsitektur <i>YOLOv4</i> (Bubbling, 2020)..... | 17 |
| Gambar 2.12. Grafik Perbandingan Kecepatan <i>YOLOv4</i> | 19 |
| Gambar 2.13. <i>Intersection over Union</i> (IoU) (Padilla dkk, 2020) | 21 |
| Gambar 2.14. Contoh Kurva <i>Precision Recall</i> (Tan, 2019)..... | 21 |
| Gambar 3.1. Tahapan Penelitian..... | 29 |
| Gambar 3.2. Contoh Citra Pada Dataset <i>CrowdHuman</i> | 30 |
| Gambar 3.3. Alur Proses Pengembangan Sistem | 32 |
| Gambar 4.1. Contoh Citra Pada Dataset <i>CrowdHuman</i> | 33 |
| Gambar 4.2. Ilustrasi Proses Konversi File Anotasi..... | 34 |
| Gambar 4.3. Pohon Direktori pengembangan Model | 35 |
| Gambar 4.4. Ilustrasi Deteksi Manusia..... | 42 |
| Gambar 4.5. Ilustrasi Transformasi Perspektif..... | 43 |
| Gambar 4.6. Kalkulasi Jarak Antara Dua Objek. | 44 |
| Gambar 4.7. Error pada Deteksi <i>Social Distancing</i> | 48 |
| Gambar 4.8. Pemberian Titik Penanda | 49 |
| Gambar 4.9. Transformasi Perspektif..... | 49 |
| Gambar 4.10. Perhitungan Jarak dan Konversi Piksel..... | 50 |

DAFTAR TABEL

| | |
|---|----|
| Tabel 2.1. Penelitian Terkait | 26 |
| Tabel 4.1. Konfigurasi Hyperparameter YOLOv4 | 37 |
| Tabel 4.2. Evaluasi Skor Model | 39 |
| Tabel 4.3. Hasil Pengujian Model Pada Citra | 39 |
| Tabel 4.4. Hasil Pengujian Model Pada Video Digital | 41 |
| Tabel 4.5. Hasil Deteksi <i>Social Distancing</i> pada <i>Bird eye View</i> | 46 |
| Tabel 4.6. Hasil Deteksi <i>Social Distancing</i> pada <i>Video</i> | 47 |
| Tabel 4.7. Pengujian Sistem Deteksi dengan Konversi Piksel | 51 |

BAB I PENDAHULUAN

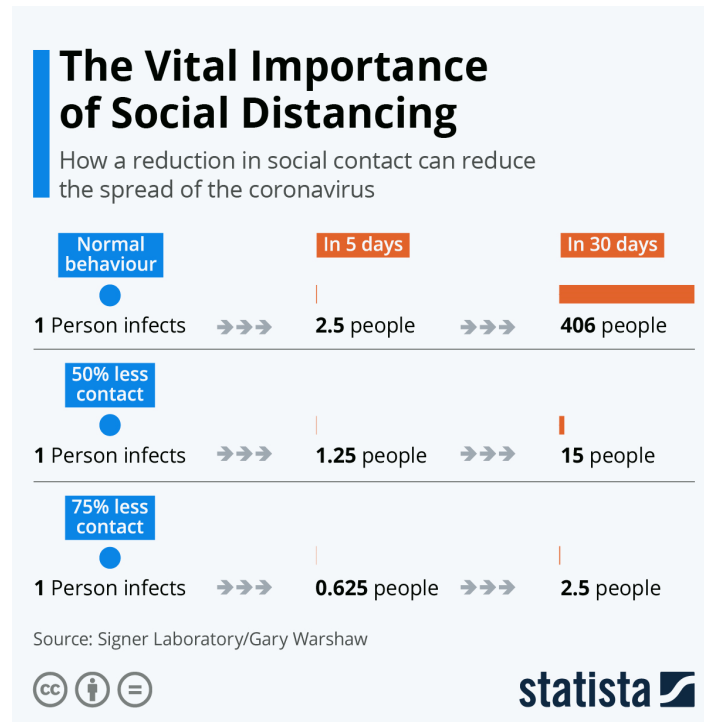
1.1. Latar Belakang

Pandemi COVID-19 (*Coronavirus Disease-19*) dianggap sebagai ancaman kesehatan masyarakat terbesar saat ini. Penyakit COVID-19 menyerang sistem pernapasan secara akut dengan gejala seperti demam, kelelahan, batuk kering dan sesak napas (Liu dkk, 2020). COVID-19 pertama kali diketahui pada Desember 2019 di Wuhan, China (Huang dkk, 2020). Berdasarkan Worldometer (2020), per 5 Februari 2021 penyakit COVID-19 ini dilaporkan telah menyebar ke 219 negara dengan total penderita aktif sebanyak 25 juta lebih orang. Lebih dari 2 juta orang meninggal dunia karena COVID-19. Penyakit ini menjadi ancaman bagi seluruh dunia karena dapat mengganggu bahkan menghancurkan berbagai sektor kehidupan. Situasi ini memaksa seluruh komunitas global untuk mencari solusi untuk mencegah penyebaran dari virus ini. Beberapa organisasi kesehatan dan ahli medis dan peneliti mencoba untuk mengembangkan obat dan juga vaksin untuk virus ini. Untuk saat ini solusi alternatif untuk menghentikan penyebaran virus ini salah satunya adalah *social distancing*.

Sebab virus ini sebagian besar menyebar kepada orang yang melakukan kontak dekat dengan orang yang terinfeksi (dengan jarak 6 kaki) dalam waktu yang lama (CDC, 2020). Virus ini juga menyebar ketika orang yang terinfeksi bersin, batuk, atau berbicara, droplets yang keluar dari hidung atau mulut menyebar melalui udara dan mengenai orang-orang sekitar. Penelitian yang baru-baru ini dilakukan memperlihatkan bahwa orang yang tidak memiliki gejala tapi terinfeksi virus, juga ambil peran dalam penyebaran virus (WHO, 2020). Oleh karena itu, sangat penting untuk menjaga jarak setidaknya 6 kaki dari orang lain, walaupun orang tersebut tidak memiliki gejala.

Social distancing merupakan langkah penanggulangan penyebaran virus, dengan meminimalkan kontak fisik dengan manusia, contohnya seperti menghindari massa tempat umum (misalnya; pusat perbelanjaan, taman, sekolah, universitas, bandara, dan tempat kerja), menghindari keramaian, dan menjaga jarak aman dengan orang-orang (CDC, 2020).

Dengan mengurangi kontak sosial dapat mengurangi penularan virus dari orang yang terinfeksi ke orang sehat, penyebaran, seperti yang ditampilkan **Gambar 1.1** (Armstrong, 2021).



Gambar 1.1. Pentingnya *Social Distancing* (Armstrong, 2021)

Pada **Gambar 1.1.** Menunjukkan ilustrasi perbandingan antara orang yang melakukan kebiasaan sosial normal dengan orang mengurangi kontak sosial dengan orang-orang sebanyak 25% dan 75%, dalam ilustrasi tersebut menunjukkan dengan melakukan *social distancing* dan mengurangi kontak sosial dengan orang-orang dapat mengurangi penyebaran dari virus COVID-19 (Armstrong, 2021).

Di Indonesia sendiri, virus COVID-19 muncul pada awal bulan Maret 2020, Presiden RI Joko Widodo meminta masyarakat Indonesia untuk melakukan *social distancing*. Di Indonesia berdasarkan laman <https://covid19.go.id/peta-sebaran> yang diakses pada tanggal 6 Februari 2021, kasus aktif virus penyakit COVID-19 adalah 176,433 orang. Lebih dari 30 ribu orang meninggal dunia karena COVID-19. Adapun tindakan yang dilakukan pemerintah untuk menerapkan *social distancing*, ialah melakukan pembatasan jumlah penumpang dalam moda transportasi umum, dilarang melakukan atau membuat suatu kerumunan, memberi

jarak pada bangku tunggu di tempat pelayanan umum seperti puskesmas, samsat, dan lain-lain dengan memberi tanda silang (X) pada setiap jarak satu kursi. Walaupun pemerintah telah memberikan aturan mengenai *social distancing*, akan sangat sulit untuk memastikan orang-orang untuk mengikuti aturan *social distancing* yang sangat krusial ini. Oleh sebab itu, dibutuhkannya sebuah sistem otomatis yang dapat mendeteksi orang yang melanggar aturan *social distancing*.

Dengan perkembangan teknologi terutama dalam menggunakan metode kecerdasan buatan dalam bidang *Computer Vision* bisa membantu melakukan deteksi manusia pada sebuah video digital ataupun citra digital. Sehingga metode ini, khususnya *deep learning* dapat digunakan untuk membangun sistem pendeteksi pelanggar *social distancing*.

Beberapa penelitian yang terkait dengan deteksi objek menggunakan algoritma deep learning sudah banyak dilakukan. Contohnya Penelitian yang dilakukan oleh Sisco Jupiyandi dkk. (2019) dengan judul “*Pengembangan Deteksi Citra Mobil untuk Mengetahui Jumlah Tempat Parkir Menggunakan CUDA dan Modified YOLO*”. Pada penelitian ini digunakan model *Modified YOLO (M-YOLO)* dengan bantuan komputasi *CUDA*, untuk membangun sebuah sistem yang dapat mengetahui jumlah slot pada lahan parkir. Hasil dari penelitian ini menunjukkan bahwa dengan menggunakan GPU dibandingkan dengan CPU dapat mempercepat waktu komputasi rata-rata sebesar 0,179 detik dengan rata-rata akurasi sebesar 100%. Dan ada juga Penelitian melakukan deteksi pejalan kaki yang dilakukan oleh Renjira Naufhal Dhiaegana (2020) dengan judul “*Penerapan Convolutional Neural Network Untuk Deteksi Pedestrian pada Sistem Autonomous Vehicle*”. Pada penelitian digunakan Model *YOLO* versi empat serta *YOLO* versi empat *tiny* untuk melakukan deteksi pejalan kaki, model pada penelitian ini dilatih dengan dataset *CrowdHuman*. Dari hasil pengujian performa model yang paling optimal adalah *YOLOv4-tiny-416-double* dengan nilai mAP 69,02.

Berdasarkan uraian diatas, pada tugas akhir ini akan dibangun sebuah sistem yang dapat mendeteksi Pelanggaran *social distancing* dengan menggunakan Algoritma *Object Detection YOLOv4* dan *Euclidean distance*. Dengan penelitian

ini diharapkan dapat memberikan hasil yang terbaik sehingga dapat digunakan untuk mengurangi kasus COVID-19.

1.1. Rumusan Masalah

Berdasarkan latar belakang penelitian yang telah dijelaskan, maka rumusan masalah pada penelitian ini adalah:

1. Bagaimana cara mengimplementasikan *YOLOv4* dan *euclidean distance* untuk membuat sistem yang dapat mendeteksi orang-orang yang sedang melakukan pelanggaran *social distancing*?
2. Bagaimana kinerja model dalam mendeteksi manusia menggunakan dataset *CrowdHuman*?
3. Bagaimana hasil sistem dalam mendeteksi pelanggar *social distancing* pada citra dan video digital?

1.2. Tujuan Penelitian

Tujuan akhir yang ingin dicapai dari penelitian ini adalah:

1. Membuat sistem yang dapat mendeteksi orang-orang yang sedang melakukan pelanggaran *social distancing* dengan mengimplementasikan *YOLOv4* dan *euclidean distance*.
2. Mengetahui kinerja model dalam mendeteksi manusia menggunakan dataset *CrowdHuman*.
3. Mengetahui hasil sistem dalam mendeteksi pelanggar *social distancing* pada citra dan video digital.

1.3. Manfaat Penelitian

Penelitian ini diharapkan dapat memberi manfaat sebagai berikut:

1. Bagi peneliti, penelitian ini dapat digunakan untuk menambah pengetahuan khususnya di bidang kecerdasan buatan untuk mendeteksi objek terkhusus manusia.
2. Bagi institusi pendidikan, penelitian ini dapat digunakan sebagai acuan dan referensi ilmiah untuk melakukan penelitian-penelitian lanjutan.
3. Bagi pemerintah maupun organisasi yang berhubungan dengan publik, penelitian ini dapat menjadi acuan ataupun referensi dalam membuat sistem yang bisa secara otomatis mendeteksi pelanggaran *social*

distancing untuk membantu pengurangan penyebaran penyakit COVID-19.

1.4. Batasan Masalah

Yang menjadi batasan masalah dalam penelitian ini adalah:

1. Objek yang dideteksi adalah manusia dalam sebuah citra dan video digital.
2. Dataset yang digunakan untuk pelatihan model adalah sekumpulan citra yang sudah dilabeli, yang bersumber dari *CrowdHuman*.
3. Sistem deteksi *social distancing* hanya melihat jarak antara orang-orang pada citra tidak mempertimbangkan jika orang yang dideteksi sedang berbicara atau sedang melakukan hal-hal lainnya.
4. Untuk data pengujian sistem deteksi pelanggaran *social distancing*, video yang digunakan bersumber dari *The Multiple Object Tracking Benchmark Challenge* dan ada juga video digital yang diambil melalui kamera *smartphone*.
5. Pada saat pengambilan data citra dan video *smartphone* jarak objek dengan camera adalah 5 meter \pm , untuk tinggi kamera dari tanah adalah 3,5 meter \pm dan terakhir sudut pengambilan gambar adalah $70^\circ \pm$.
6. Untuk data video dan juga citra yang diambil dengan kamera *smartphone* dilakukan pada siang hari sehingga objek pada citra maupun video kelihatan dengan jelas.

BAB II TINJAUAN PUSTAKA

2.1. *Social Distancing*

Social distancing atau menjaga jarak sosial, adalah salah satu langkah pencegahan dan pengendalian infeksi virus COVID-19 dengan cara menjaga jarak saat berinteraksi dengan orang lain dan menghindari tempat yang ramai. Contoh dari penerapan *social distancing* dapat dilihat pada **Gambar 2.1**.



Gambar 2.1. Praktik *Social Distancing*.

Pada **Gambar 2.1** menunjukkan praktik *social distancing* yang dilakukan di tempat umum, menurut WHO (2020) ketika menerapkan *social distancing*, seseorang tidak diperkenankan untuk berjabat tangan serta menjaga jarak setidaknya 6 kaki saat berinteraksi dengan orang lain, terutama dengan orang yang sedang sakit atau berisiko tinggi menderita COVID-19.

2.2. *Citra Digital*

Citra (*image*) secara harfiah merupakan gambar pada bidang dwimatra (dua dimensi). Sedangkan ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek kemudian memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh alat-alat optik, misalnya mata pada manusia, kamera, pemindaian (*scanner*) dan sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam (Permadi & Murinto, 2015).

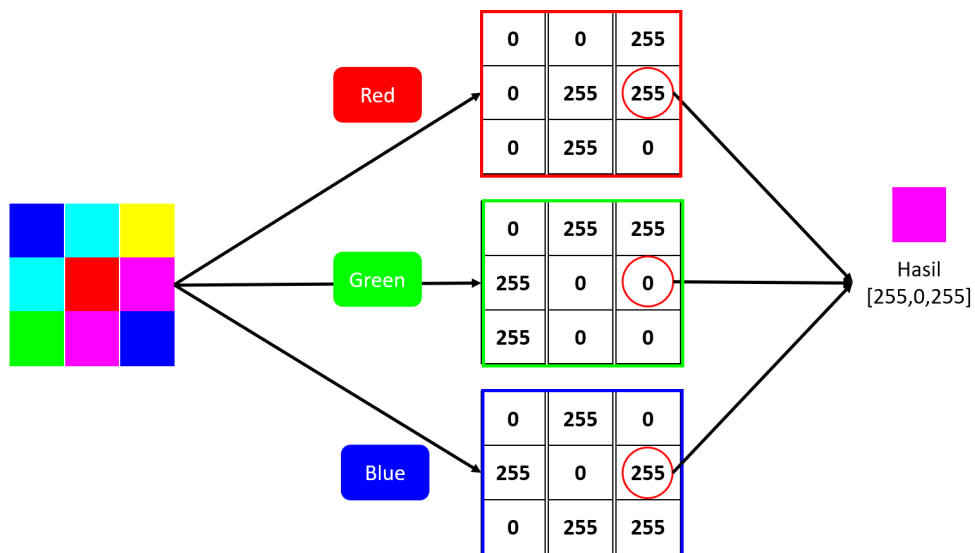
Suatu citra dapat didefinisikan sebagai fungsi $f(x,y)$ berukuran M baris dan N kolom. Jika nilai x,y dan amplitudo f berhingga dan diskrit, maka citra tersebut

dapat dikatakan citra digital. Citra digital dapat ditulis dalam bentuk matriks pada **Persamaan 2.1**.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,m-1) \\ f(1,0) & f(1,1) & & f(1,m-1) \\ & \vdots & \ddots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix} \quad (2.1)$$

Dapat dilihat pada **Persamaan 2.1** diketahui bahwa pada matrix $f(x,y)$ piksel mempunyai dua parameter, yaitu koordinat dan intensitas atau warna. Secara sistematis, citra digital dapat ditulis sebagai fungsi intensitas $f(x,y)$ dimana x (baris) dan y (kolom) merupakan koordinat posisi dan $f(x,y)$ adalah nilai fungsi pada setiap titik (x,y) yang menyatakan besar intensitas citra atau tingkat keabuan atau warna dari piksel di titik tersebut (Nalwan, 1997).

Pada citra digital berwarna biasa disebut RGB (*Red, Green, Blue*), masing masing piksel memiliki warna tertentu, warna tersebut adalah merah (*Red*), hijau (*Green*), dan biru (*Blue*). Jika masing masing warna memiliki range 0-255, maka total variasi warna berbeda pada gambar adalah $255^3 = 16.581.375$ variasi warna.



Gambar 2.2. Contoh Citra Berwarna

Pada **Gambar 2.2** dapat dilihat contoh citra berwarna dan representasi matriksnya (red, green dan blue), setiap piksel memiliki representasi nilai RGB nya masing-masing, seperti pada baris 3 kolom 2 representasi nilainya adalah 255, 0, 255.

2.3. Video Digital

Video digital adalah sekelompok gambar atau citra digital yang tersusun secara teratur & berurutan sehingga membuat efek bergerak pada objek yang berada dalam gambar atau citra digital. Video digital terdiri dari beberapa elemen, yaitu :

1. *Frame rate*, adalah jumlah *frame* yang ditampilkan tiap detik.
2. *Frame dimensions*, adalah lebar & tinggi dari *frame* video yang dinyatakan dalam ukuran piksel.
3. *Pixel depth*, adalah jumlah bit per piksel. (Manning, 2008).

2.4. Deep Learning

Deep learning merupakan sebuah subbagian bidang keilmuan pada bidang *machine learning* yang berdasarkan *Artificial Neural Network* (ANN) atau bisa juga yang dimana mengajarkan komputer agar dapat melakukan tindakan yang berulang yang dilakukan oleh manusia. Dalam Deep Learning, sebuah komputer bisa belajar untuk mengklasifikasi gambar, suara, teks, ataupun video. Dalam mengklasifikasi citra pada *Deep Learning*, pertama sebuah komputer dilatih dengan menggunakan data set berlabel dan dengan jumlah yang cukup besar yang kemudian dapat mengubah nilai piksel dari sebuah gambar menjadi suatu representasi internal atau *feature map* yang dimana penggunaan *feature map* dapat digunakan untuk mendeteksi atau mengklasifikasi pola pada masukan input. (Tuluran, 2020).

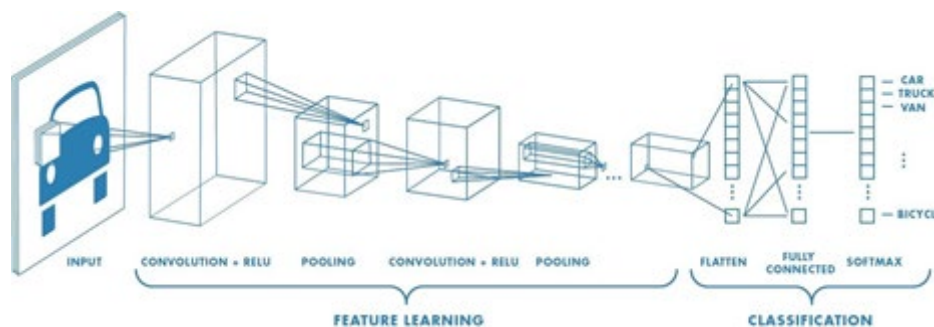
Metode pendekatan deep learning mengklasifikasi data dalam dua sesi, yaitu sesi *training* dan *testing*. Pada sesi *training* mempelajari ekstraksi fitur dari setiap data supaya bisa membedakan suatu label dengan label yang lain. Pada sesi *testing* data-data yang diuji dapat dianalisis dari hasil sesi *training*. Salah satu contoh dari algoritma dalam deep learning ialah *Convolutional Neural Network*

2.5. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan salah satu algoritma dari deep learning yang merupakan hasil pengembangan dari *Multilayer Perceptron* (MLP) yang dirancang untuk melakukan olah data citra. CNN dipakai untuk melakukan klasifikasi data yang berlabel dengan menggunakan metode *supervised learning* yang cara kerjanya menggunakan menggunakan data yang dilatih

memiliki label yang ditargetkan sehingga tujuan dari metode ini yaitu mengelompokkan suatu data ke data yang telah ada (Tuluran, 2020).

CNN hanya terbatas untuk memproses data yang memiliki struktur kotak (*grid*), contohnya adalah citra digital. Konvolusi adalah operasi dari aljabar linear yang mengalikan matriks *filter* dengan citra yang nantinya akan diproses. Proses ini terjadi pada lapisan konvolusi dan merupakan salah satu jenis dari banyak lapisan yang bisa dimiliki dalam suatu jaringan CNN. Meskipun begitu, lapisan konvolusi ini adalah lapisan yang paling penting dalam mengkalsifikasikan citra. Jenis lapisan yang lain yang biasa digunakan adalah *Pooling Layer*, lapisan ini digunakan untuk mencari suatu nilai maksimum atau nilai rata-rata dari bagian-bagian tertentu pada lapisan piksel pada citra masukan. Adapun gambaran dari arsitektur *Convolution Neural Network* dapat dilihat pada **Gambar 2.3** (Tuluran, 2020).



Gambar 2.3. Proses *Convolutional Neural Network*

Pada **Gambar 2.3** setiap lapisan masukan atau input mempunyai susunan neuron 3 dimensi, yaitu (lebar, tinggi, dan kedalaman) lebar dan tinggi merupakan ukuran lapisan, lalu kedalaman mengacu pada jumlah lapisan. Model jaringan CNN ini digunakan untuk melakukan klasifikasi citra. Sebuah *convolution neural network* dapat memiliki banyak lapisan-lapisan yang masing-masing lapisan mempelajari bentuk dari objek pada citra. Pengolahan citra dilakukan pada setiap data latih dan *output* dari masing-masing data gambar yang diolah lalu digunakan sebagai masukan ke lapisan berikutnya. Pengolahan citra dapat dimulai sebagai *feature* yang sederhana, seperti ukuran kecerahan dan tepi atau meningkatkan kompleksitas pada fitur secara unik untuk menentukan objek sesuai ketebalan lapisan (Putri, 2018).

Secara garis besar, Terdapat beberapa proses yang terjadi di *convolution neural network* yaitu:

2.6.1. Convolutional Layer

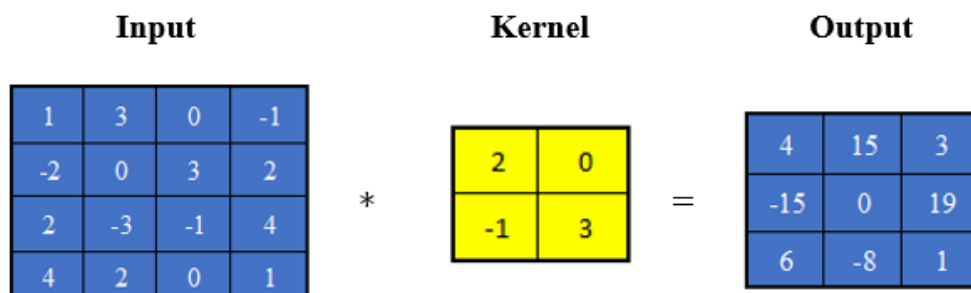
Convolution layer merupakan proses utama yang mendasari jaringan arsitektur CNN dan terdiri atas *kernel*. *Kernel-kernel* pada lapisan ini disebut filter konvolusi. *Kernel* berfungsi mempelajari fitur-fitur lokal pada feature map. Tahap convolutional layer melakukan operasi konvolusi pada output dari layer sebelumnya. Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Persamaan konvolusi merupakan persamaan pada dua fungsi argument bernilai riil. Operasi konvolusi $g(x)$ dapat ditunjukkan pada **Persamaan 2.2**:

$$g(x) = w(x) * f(x) \tag{2.2}$$

Dimana $f(x)$ adalah input atau citra asli dan $w(x)$ adalah *kernel*. *Input convolution layer* merupakan gambar yang direpresentasikan menjadi sebuah matriks. Operasi konvolusi menghasilkan nilai tinggi dan rendah pada posisi tertentu pada *feature map*. Posisi tertentu dari konvolusi kernel, merupakan perkalian untuk setiap nilai pada sel *kernel* dan nilai piksel gambar yang tumpang tindih dengan sel *kernel* (Khan dkk, 2018). Adapun operasi perkaliannya menggunakan **Persamaan 2.3**:

$$h_{i,j} = \sum_{k=1}^m \sum_{l=1}^m W_{k,l} X_{i+k-1,j+l-1} \tag{2.3}$$

Dimana m adalah lebar dan tinggi *kernel*, h adalah *input convolution*, X adalah *input*, dan W adalah *convolutional kernel*. Sebagai contoh, ilustrasi pada proses *convolution layer* dengan nilai *stride* 2 dapat dilihat pada **Gambar 2.4**.



Gambar 2.4. Ilustrasi Proses Convolutional Layer

Adapun perhitungan nilai pada input convolution yang diperoleh pada **Gambar 2.4** dengan menggunakan **Persamaan 2.3**.

$$h_{1,1} = (1 \times 2) + (3 \times 0) + (-2 \times -1) + (0 \times 3) = 4$$

$$h_{1,2} = (3 \times 2) + (0 \times 0) + (0 \times -1) + (3 \times 3) = 15$$

$$h_{1,3} = (0 \times 2) + (-1 \times 0) + (3 \times -1) + (2 \times 3) = 3$$

$$h_{2,1} = (-2 \times 2) + (0 \times 0) + (2 \times -1) + (-3 \times 3) = -15$$

$$h_{2,2} = (0 \times 2) + (3 \times 0) + (-3 \times -1) + (-1 \times 3) = 0$$

$$h_{2,3} = (3 \times 2) + (2 \times 0) + (-1 \times -1) + (4 \times 3) = 19$$

$$h_{3,1} = (2 \times 2) + (-3 \times 0) + (4 \times -1) + (2 \times 3) = 6$$

$$h_{3,2} = (-3 \times 2) + (-1 \times 0) + (2 \times -1) + (0 \times 3) = -8$$

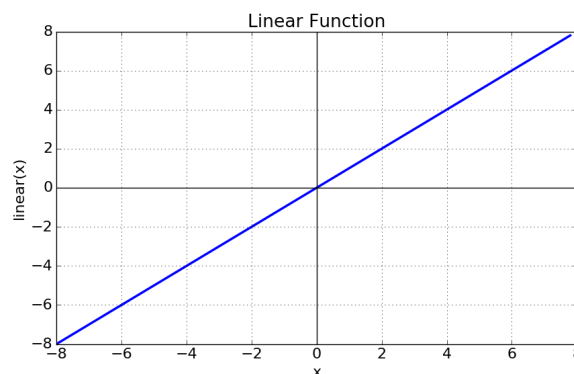
$$h_{3,3} = (-1 \times 2) + (4 \times 0) + (0 \times -1) + (1 \times 3) = 1$$

2.6.2. Fungsi Aktivasi

Fungsi aktivasi merupakan operasi matematik yang dikenakan pada sinyal output y . Fungsi aktivasi berfungsi menentukan apakah suatu *neuron* aktif atau tidak berdasarkan *weighted sum* dari suatu *input*. Berikut ini merupakan fungsi aktivasi yang digunakan pada penelitian ini:

a. Fungsi Aktivasi Linear

Fungsi Aktivasi Linear merupakan fungsi aktivasi yang nilai input nya sama dengan nilai outputnya. Adapun grafik dari fungsi aktivasi linear dapat dilihat pada **Gambar 2.5**.



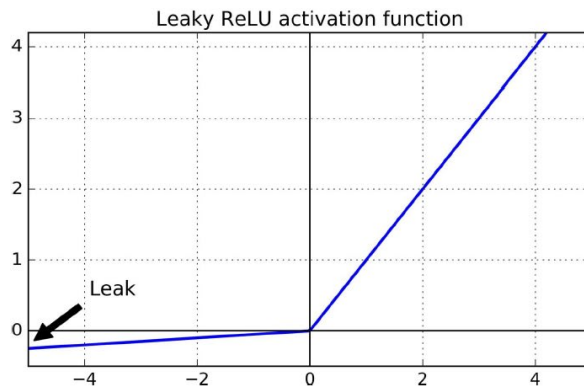
Gambar 2.5. Fungsi Aktivasi Linear

Berdasarkan **Gambar 2.5** fungsi aktivasi linear adalah fungsi aktivasi yang sangat sederhana yang dimana nilai *input* sama dengan *output* nya. Adapun rumus matematis dari fungsi aktivasi linear dapat dilihat pada **Persamaan 2.4**.

$$f(x) = x \quad (2.4)$$

b. Fungsi Aktivasi *Leaky ReLU*

Fungsi Aktivasi *Leaky Rectifier Linear Unit (Leaky ReLU)* merupakan pengembangan dari fungsi aktivasi *ReLU*. *Leaky ReLU* menambahkan sebuah *Leak* dan memberlebar nilai output dari neuron. Adapun grafik dari fungsi aktivasi *Leaky ReLU* dapat di lihat pada **Gambar 2.6** (Maas dkk, 2013).



Gambar 2.6. Fungsi Aktivasi *Leaky ReLU*

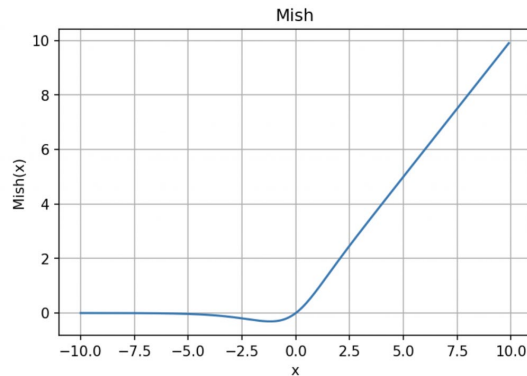
Bisa dilihat pada **Gambar 2.6** *Leaky ReLU* membuat limit jika nilai inputan dibawah 0 dimana nilai input dikalikan dengan 0,01 dan jika nilai input ≤ 0 maka nilai input sama dengan nilai output. Adapun rumus matematis dari *Leaky ReLU* dapat dapat dilihat **Persamaan 2.5** (Maas dkk, 2013).

$$f(x) = \begin{cases} 0.01x, & \text{if } x < 0 \\ x, & \text{otherwise} \end{cases} \quad (2.5)$$

c. Fungsi Aktivasi *Mish*

Fungsi Aktivasi *Mish* adalah salah satu fungsi aktivasi yang dikembangkan oleh Diganta Misra dalam penelitiannya yang berjudul “*Mish: A Self Regularized Non-Monotonic Activation Function*”, *Mish function* memiliki performa yang lebih bagus dibandingkan dengan *ReLU* tetapi

memerlukan komputasi yang lebih banyak. Adapun grafik dari fungsi aktivasi *mish* dapat dilihat pada **Gambar 2.7**.



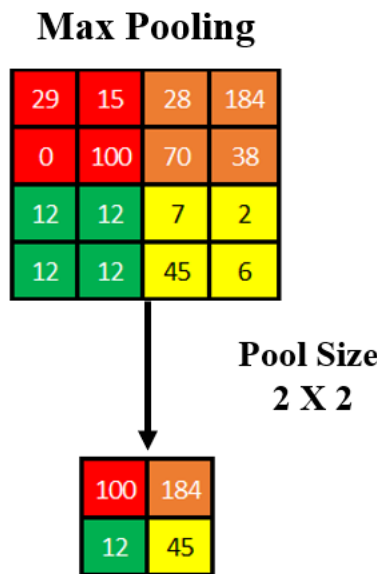
Gambar 2.7. Fungsi Aktivasi *Mish*

Pada **Gambar 2.7** bisa dilihat grafik input dan output dari fungsi aktivasi *mish* adapun rumus matematis dari Fungsi aktivasi Mish dapat dilihat pada **Persamaan 2.6** (Misra, 2019).

$$f(x) = x \cdot \tanh(\ln(1 + e^x)) \quad (2.6)$$

2.6.3. Pooling Layer

Pooling layer terletak setelah *convolution layer*. Pada *pooling layer* dilakukan proses pooling. Fungsi dari pooling ini adalah untuk mengurangi jumlah parameter dengan operasi down-sampling. Jenis *pooling layer* yang biasa digunakan yaitu *average pooling* dan *max pooling*. Nilai yang diambil pada *average pooling* adalah nilai rata-rata dan pada *max pooling* yang diambil adalah nilai maksimal. Lapisan *pooling* yang dimasukkan di antara lapisan konvolusi pada model CNN dapat mengurangi ukuran volume output pada feature map, sehingga mengurangi jumlah parameter dan perhitungan di jaringan. Adapun proses dari *max pooling* dapat dilihat pada **Gambar 2.8**.

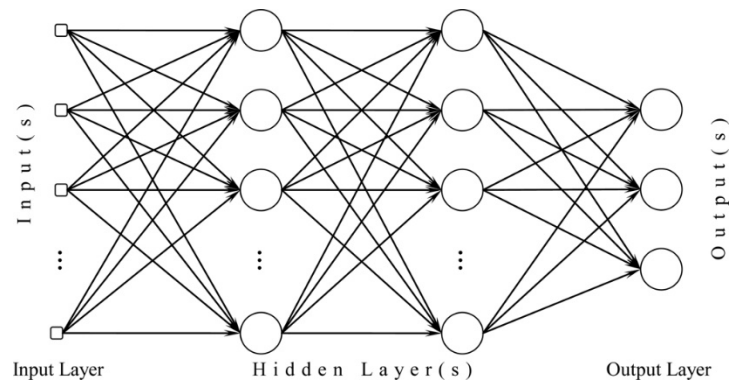


Gambar 2.8. *Pooling Layer*

Pada **Gambar 2.8** merupakan proses dari *max pooling* dengan *pool size 2x2* dan *stride 2*, dan *output* dari proses pooling merupakan sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan citra awal.

2.6.4. Fully Connected Layer

Setelah melewati *pooling layer*, Hasil dari pooling layer dimasukkan untuk *Fully connected layer* merupakan sebuah lapisan dimana semua neuron aktivasi dari lapisan sebelumnya terhubung dengan neuron lapisan selanjutnya sama seperti halnya dengan neural network biasa. Perbedaan antara lapisan *Fully-Connected* dan lapisan konvolusi biasa adalah *neuron* di lapisan konvolusi terhubung hanya ke daerah tertentu pada input, sementara lapisan *Fully-Connected* memiliki *neuron* yang secara keseluruhan terhubung (Nurhikmat, 2018). Pada **Gambar 2.9** diperlihatkan ilustrasi dari fully-connected layer.



Gambar 2.9. Ilustrasi *Fully Connected Layer*

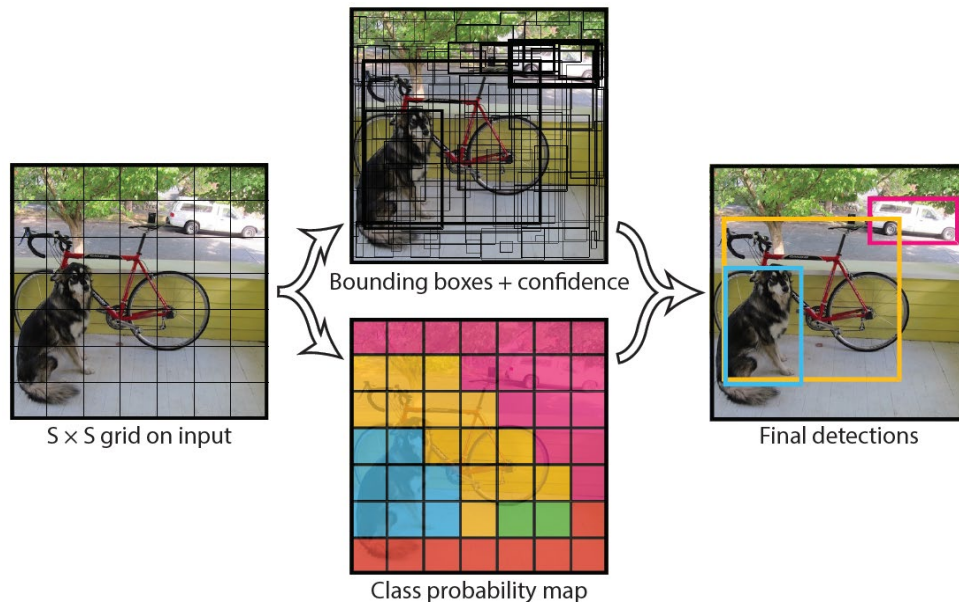
Gambar 2.9 merupakan lapisan *fully connected layer*, dimana nilai inputnya diperoleh dari hasil dari *flattening*. Lapisan *fully connected layer* disebut juga sebagai *hidden layer*, dimana banyaknya jumlah *hidden layer* tidak ada batasan ataupun bebas dalam menentukannya. Banyaknya jumlah *neurons* pada setiap lapis dari *hidden layer* juga tidak ada batasan ataupun bebas dalam menentukannya, tetapi dalam hal ini harus disesuaikan dengan hardware komputer. Karena semakin banyak jumlah *hidden layer* dan neuronnya, maka proses komputasinya akan semakin lama dan berat.

2.6. You Only Look Once (YOLO)

YOLO atau *You Only Look Once* adalah salah satu pendeteksi objek *single-stage* selain SSD (*Single Shot Detector*). YOLO adalah pendekatan baru dalam ranah deteksi objek. Dengan YOLO, deteksi objek dilakukan dengan melihat masalah tersebut sebagai masalah regresi untuk memisahkan secara spasial *bounding box* dan kelas probabilitas yang terkait terhadap *bounding box* tersebut. Sebuah *neural network* memprediksi *bounding box* dan kelas prediksi langsung dari keseluruhan citra dari satu evaluasi.

YOLO mempunyai sejumlah kelebihan dibandingkan sistem berbasis *classifier*. YOLO melihat pada keseluruhan citra pada saat pengujian sehingga prediksinya diinformasikan oleh konteks global dalam gambar. YOLO juga menghasilkan prediksi dengan jaringan evaluasi tunggal tidak seperti sistem R-CNN yang membutuhkan banyak jaringan untuk memproses satu gambar. Ini

membuat YOLO lebih cepat, bahkan ribuan kali lebih cepat dibandingkan R-CNN dan ratusan kali lebih cepat dari Fast R-CNN (Redmon, 2018).

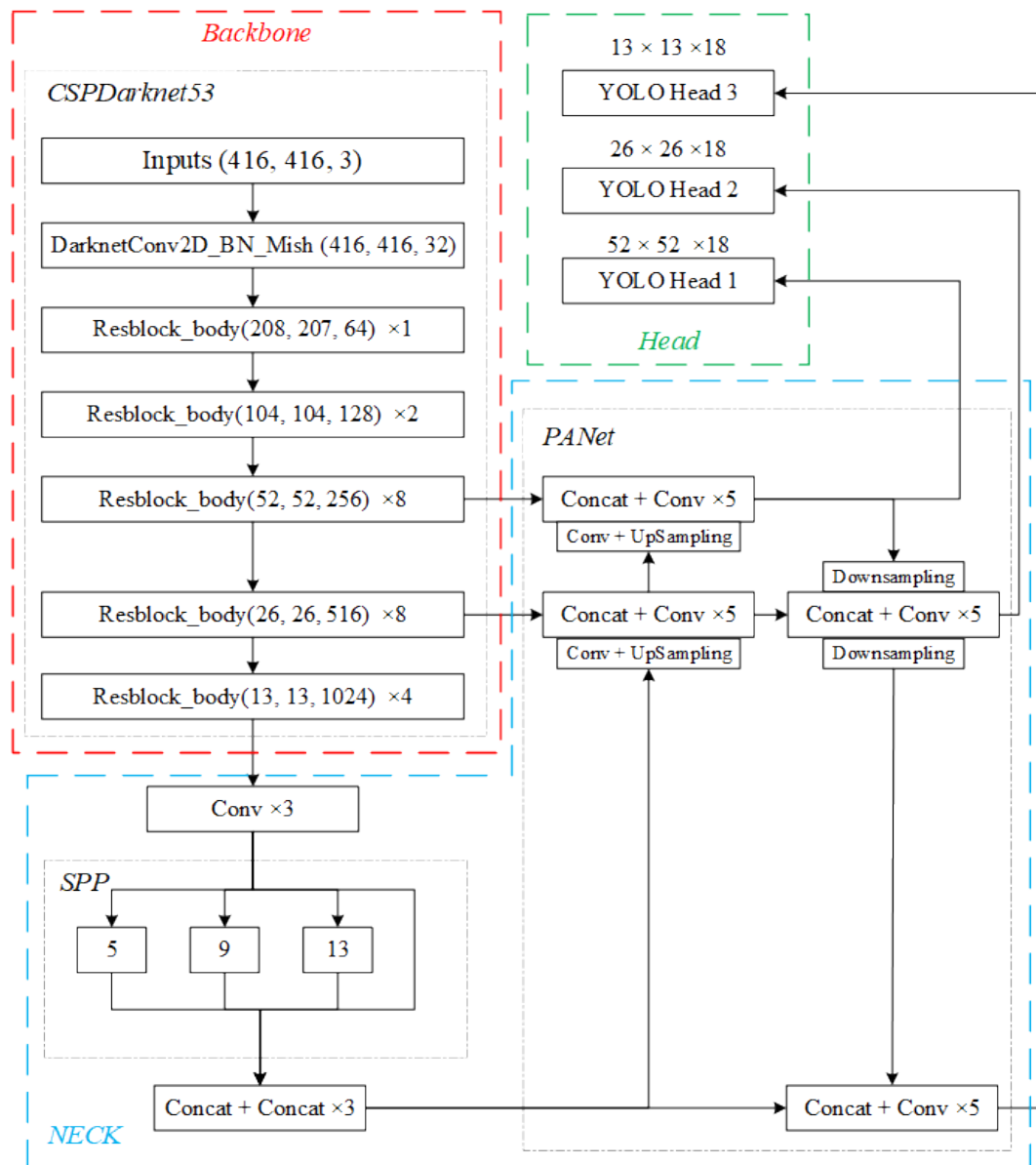


Gambar 2.10. Ilustrasi Cara Kerja YOLO (Redmon dkk, 2016)

Pada **Gambar 2.10**. Pada awalnya citra akan dipecah dalam suatu *grid* berdimensi $S \times S$. Setiap *cell* pada *grid* tersebut akan diprediksi N kemungkinan *Bounding boxes* dan nilai probabilitasnya. Kebanyakan dari *Bounding box* tersebut memiliki probabilitas yang sangat rendah, sehingga algoritma ini akan menghapus *box – box* tersebut yang memiliki probabilitas dibawah batas tertentu. *Box* hasil dari *filter* tersebut akan dilanjutkan dengan proses *Non-Maximum Suppression* (NMS) sehingga diperoleh satu posisi dari objek yang paling akurat menurut metode YOLO ini (Redmon dkk, 2016).

Pada versi ketiga, YOLO melakukan beberapa perubahan dan pengembangan untuk membuat model tersebut lebih baik lagi. *Training* yang dilakukan terhadap *network* membuat model tersebut membesar (ukurannya), dengan hasil akurasi yang lebih tinggi dari sebelumnya dengan kecepatan yang tetap sama. Pada 320×320 YOLOv3 berjalan 22ms dengan 28.2 mAP, dimana memiliki akurasi yang sama seperti SSD tetapi memiliki performa yang tiga kali lebih cepat. YOLO ini pada umumnya berjalan pada *backbone network Darknet* (Redmon, 2013), dimana dengan *backbone* tersebut, model YOLO mendapatkan kecepatan pemrosesan lebih

tinggi dari *backbone* lainnya yang diuji seperti *Residual Network* (Dhiaegana, 2020).



Gambar 2.11. Arsitektur *YOLOv4* (Bubbling, 2020)

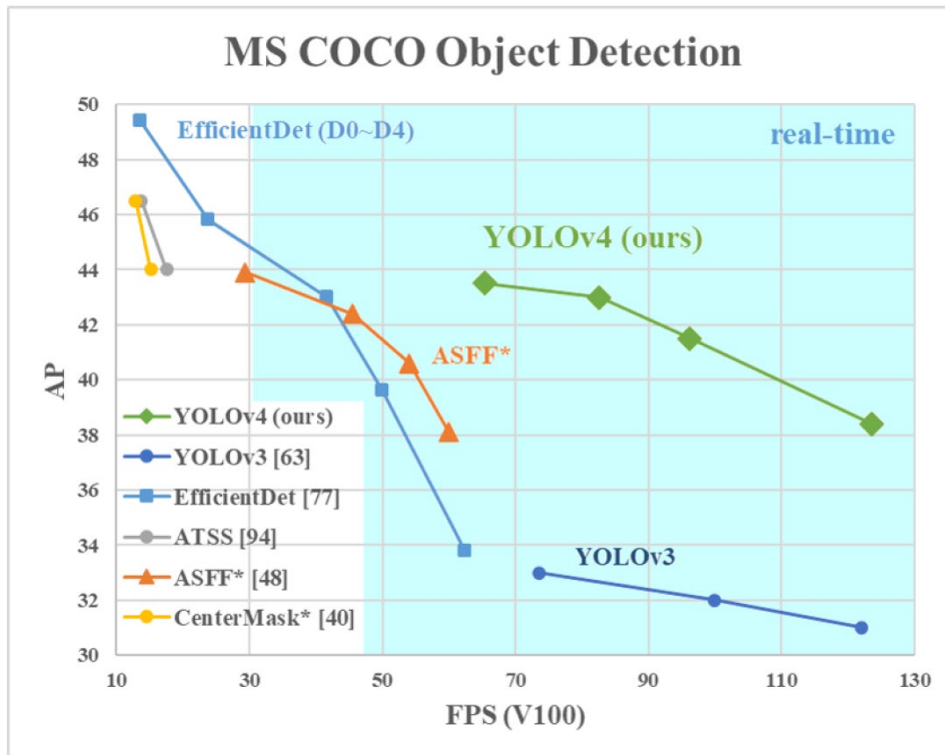
YOLO versi keempat yang dikembangkan oleh Alexey Bochkovskiy (2020) menambahkan beberapa tambahan pada jaringannya untuk meningkatkan akurasi dan efisiensi. *Weighted Residual Connection* yang mempelajari untuk mengkombinasikan residu dari lapisan neural network secara efektif dan efisien (He dkk, 2016). *Cross Stage Partial Connections* (CSP) mengintegrasikan *feature map* dari

tahap awal hingga tahap akhir jaringan. Implementasi CSP mengurangi komputasi sebanyak 20% serta mengungguli arsitektur *backbone state-of-the-art* lainnya. *Cross Mini-Batch Normalization* mengatasi masalah dimana pada statistik dihasilkan saat normalisasi didefinisikan tidak dapat diestimasi dengan baik (Yao dkk, 2020). *Self Adversarial Training* yaitu teknik augmentasi data baru yang beroperasi di 2 tahap *forward* dan *backward* pada jaringan. *Mish-activation* adalah fungsi aktivasi non-monoton yang me-regularisasi sendiri (Misra, 2019). *Mosaic Data Augmentation* adalah teknik augmentasi data dengan cara menggabungkan beberapa citra pada dataset menjadi satu. *DropBlock regularization* digunakan sebagai metode regularisasi baru untuk CNN (Ghiassi, 2018). Terakhir adalah CIOU *loss*, sebuah fungsi *loss* yang mencapai konvergensi lebih cepat dan akurasi yang lebih baik pada permasalahan regresi *bounding box* (Zheng dkk, 2020).

Pada **Gambar 2.11** Dapat dilihat Arsitektur ini menggunakan *backbone network CSPDarknet53*, untuk bagian *Neck* menggunakan SPP (*Spatial Pyramid Pooling*) dan PANet (*Path Aggregation Network*), dan untuk bagian *head* terdiri dari 3 *YOLO head* yang masing-masing menghasilkan *feature maps* berukuran $S^2 \times (3 * (5 + C))$. Dimana S merepresentasikan ukuran grid, 3 merupakan jumlah *bounding box* yang dihasilkan per *feature map*, (5+C) merupakan atribut dari *bounding box* yaitu koordinat titik tengah *bounding box*, lebar *bounding box*, tinggi *bounding box*, *objectness score* dan C merupakan *confidence score bounding box* untuk setiap kelas karena pada penelitian ini hanya ada 1 kelas yaitu *human* maka C bernilai 1.

Dapat dilihat pada **Gambar 2.11** Bagian-bagian pada *YOLOv4* melakukan tugas berbeda-beda, pada bagian *backbone CSPdarknet53* melakukan ekstraksi fitur pada citra. Pada bagian *neck* ada SPP melakukan *max pooling* dengan 4 *pool size* yang berbeda-beda (5×5 , 9×9 , 13×13 , 1×1), output dari 4 proses *max pooling* tersebut akan digabungkan dan selanjutnya akan diproses pada PANet. PANet bertugas meningkatkan kualitas dari *feature map* dengan cara melakukan *upsampling* dan *downsampling* dengan menggunakan *low-level feature map* dan *high-level feature map*. Pada bagian *head* melakukan proses deteksi objek dengan 3 ukuran berbeda *YOLO head* pertama bertugas dalam mendeteksi objek yang

berukuran kecil, YOLO *head* kedua bertugas dalam mendeteksi objek yang berukuran sedang, dan YOLO *head* ketiga bertugas dalam mendeteksi objek dengan ukuran yang besar. Ketiga proses deteksi ini terjadi pada Layer (140, 151, dan 162). Perbandingan kecepatan *YOLOv4* dengan arsitektur lainnya dapat dilihat pada **Gambar 2.12**.



Gambar 2.12. Grafik Perbandingan Kecepatan *YOLOv4*

Pada **Gambar 2.12**. Bisa dilihat perbandingan kecepatan Algoritma *YOLOv4* dengan algoritma *object detection* lainnya, dalam mendeteksi *dataset MS COCO*. Algoritma *YOLOv4* pada grafik tersebut dapat mendeteksi dengan skor AP yang baik dengan komputasi yang cepat, sehingga dapat menghasilkan prediksi baik dan *realtime*.

2.7. Metrik Perhitungan Performa Model *Object Detection*

Model yang dihasilkan dengan proses machine learning memang dapat memprediksi semua keluaran. Tetapi, hasil dari keluaran tersebut harus memiliki metrik yang menyatakan tingkat keberhasilan sebuah model mempelajari suatu data yang ada. Untuk CNN, terdapat sejumlah metrik yang digunakan untuk menghitung

performa dari model tersebut. Pada penelitian ini metrik yang digunakan adalah *mean average precision* (mAP) dan *Frame Per Second* (FPS).

2.10.1. Mean Average Precision (mAP)

Salah satu metrik yang paling umum digunakan dalam permasalahan pendeteksian objek adalah metrik *Average Precision* (AP). Secara ringkas, *average precision* menghitung nilai rata-rata *precision* untuk setiap nilai *recall* yang berada pada rentang 0 hingga 1. *Precision* adalah nilai yang menentukan seberapa akurat prediksi yang dilakukan. *Precision* dapat dinyatakan dengan **Persamaan 2.7**.

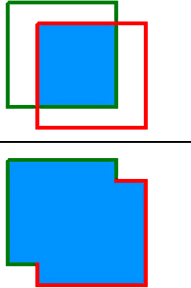
$$Precision = \frac{TP}{TP + FP} \quad (2.7)$$

Dimana TP adalah *true positive* untuk nilai aktual positif dan hasil prediksi positif, dan FP adalah *false positive* untuk nilai aktual negatif yang diprediksi positif. *Recall* menandakan seberapa baik model yang dihasilkan mencari nilai positif. *Recall* dapat dinyatakan dengan **Persamaan 2.8**.

$$Recall = \frac{TP}{TP + FN} \quad (2.8)$$

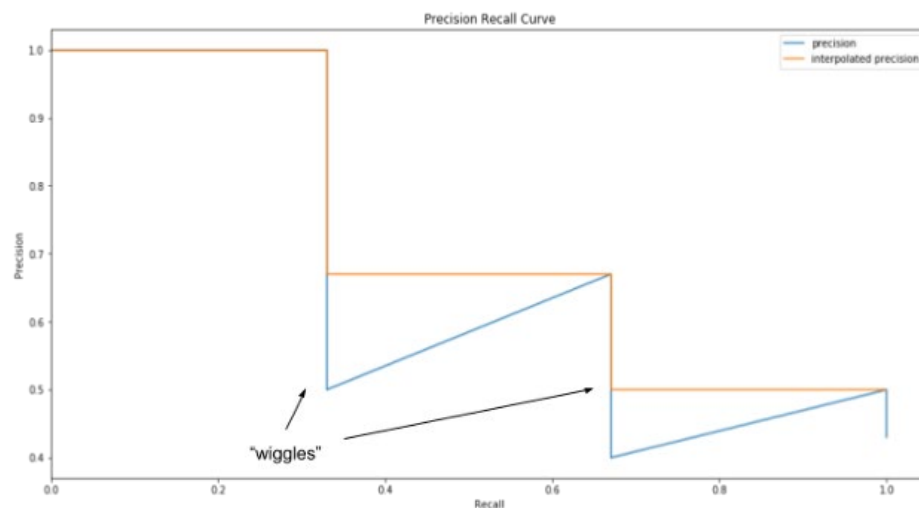
Dimana FN adalah *false negative* yang berarti nilai aktual negatif yang diprediksi sebagai nilai positif. Sebagai ringkasan, nilai *precision* adalah jumlah prediksi nilai positif yang benar dibagi dengan jumlah keseluruhan nilai yang diprediksi sebagai nilai positif. Nilai *recall* adalah jumlah prediksi positif yang benar dibagi dengan jumlah nilai positif keseluruhan sebenarnya (Pedregosa dkk, 2011).

Pada *object detection* untuk menghitung presisi pada sebuah citra, diperlukan perhitungan *Intersection over Union*. *Intersection over Union* dapat dilihat pada **Gambar 2.13**.

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of overlap}}{\text{area of union}}$$


Gambar 2.13. *Intersection over Union (IoU)* (Padilla dkk, 2020)

Dapat dilihat pada **Gambar 2.13** *Intersection over Union (IoU)* menghitung seberapa banyak wilayah prediksi *bounding box* yang beririsan dengan *bounding box* sebenarnya. IoU pada umumnya memiliki batasan (*threshold*) yang menentukan apakah sebuah prediksi bernilai *true positive* atau *false positive* (Dhiaegana, 2020).



Gambar 2.14. Contoh Kurva *Precision Recall* (Tan, 2019)

Cara menghitung *average precision* pada sebuah model adalah dengan mencari luas dibawah kurva *precision-recall* seperti pada **Gambar 2.14**. Kurva *precision-recall* yang pada umumnya membentuk pola zig-zag, dihaluskan dengan cara menukar nilai *precision* dengan nilai *precision* maksimum paling kanan dari nilai *recall*. Setelah melakukan pelembutan pola zig-zag yang ada pada kurva *precision-recall*, hitung nilai luas yang ada dibawah kurva (Dhiaegana, 2020). Rumus matematis dari *average precision* dapat dilihat pada **Persamaan 2.9**.

$$AP = \sum_{i=0}^n [Recall_{i+1} - Recall_i] * P_{inter}(Recall_{i+1}) \quad (2.9)$$

Dengan

$$P_{inter}(r) = \max_{\tilde{r} \geq r} p(\tilde{r}) \quad (2.10)$$

Untuk perhitungan *mean average precision* atau mAP, hasil perhitungan *average precision* untuk tiap kelas dirata-ratakan sesuai jumlahnya. Khususnya untuk penelitian ini, dikarenakan hanya ada satu kelas yang akan dideteksi, yakni *human*, alhasil nilai *mean average precision* sama dengan nilai *average precision*.

2.10.2. Frame Per Second

Pendeteksian pelanggaran *social distancing* harus dilakukan secara *real-time* agar sistem dapat mendeteksi dengan tepat dalam waktu yang singkat. Mata manusia dan penerima data dan sistem transmisinya dapat membentuk, mentransmisikan, dan menganalisis 10 hingga 12 citra per detik. Pusat pengaturan penglihatan yang berada di otak menyimpan setiap citra individu untuk seperlima belas detik. Jika pusat otak menerima citra lain saat seperlima belas detik, maka mekanisme penglihatan akan membuat sensasi kontinuitas visual (Read, 2000).

Terdapat tiga komponen utama dalam keseluruhan proses yang menggambarkan sistem *real-time*. Waktu adalah sumber daya paling berharga pada sistem *real-time*. Pekerjaan harus diberikan dan dijadwalkan untuk diselesaikan sebelum tenggat waktunya. Apabila hal ini di analogikan dengan deteksi pelanggaran *social distancing*, maka tugas untuk mendeteksi orang-orang harus diselesaikan dengan waktu yang cepat. Kecepatan yang dihitung pada proses pendeteksian ini adalah seberapa banyak *frame* yang dapat diproses tiap detiknya (*frame per second*). Selain waktu, keandalan juga menjadi hal yang penting, karena kegagalan sistem *real-time* dapat menyebabkan kerugian. Ketiga, lingkungan dimana sistem tersebut beroperasi adalah komponen aktif (Shin & Ramanathan, 1994).

2.8. *Euclidean Distance*

Dalam matematika *euclidean distance* antara dua titik dalam ruang *euclidean* adalah panjang ruas garis antara dua titik. *Euclidean distance* dapat dihitung dari titik koordinat *cartesian*-nya menggunakan teorema pythagoras. Misalkan, titik p memiliki koordinat *Cartesian* (p_1, p_2) dan misalkan titik q mempunyai titik koordinat (q_1, q_2) , maka jarak antara titik p dan q dapat dilihat pada **Persamaan 2.11** (Cohen, 2004).

$$d(p, q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2} \quad (2.11)$$

2.9. Transformasi Pespektif

Transformasi perspektif digunakan untuk mengubah proyeksi perspektif menjadi proyeksi parallel (Geetha & Murali, 2013). Transformasi perspektif digunakan untuk mengubah tampilan citra. Transformasi perspektif dapat dinyatakan dengan **Persamaan 2.12**:

$$p' = H \cdot p \quad (2.12)$$

Dimana p' merupakan titik koordinat destinasi transformasi citra, p merupakan titik citra sebelum di transformasi dan H merupakan matriks Homografi. Dalam bentuk matriks, Transformasi perspektif dapat dinyatakan dengan **Persamaan 2.13**.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2.13)$$

Dimana x' dan y' merupakan titik koordinat destinasi dan x dan y titik awal citra, dan a_1 hingga a_8 merupakan matriks homografi. Transformasi Perspektif juga dapat dinyatakan dengan **Persamaan 2.14**.

$$\begin{aligned} x' &= (a_1x + a_2y + a_3)/(a_7 + a_8y + 1) \\ y' &= (a_4x + a_5y + a_6)/(a_7 + a_8y + 1) \end{aligned} \quad (2.14)$$

Dengan menggunakan **Persamaan 2.13** dan **2.14**, maka akan diperoleh nilai a_1 hingga a_8 .

2.10. Penelitian Terdahulu

Berdasarkan penelitian yang akan dilakukan tentang *deep learning* dan *YOLOv4*, referensi dari beberapa penelitian terdahulu menjadi sangat krusial dalam melakukan penelitian dengan tujuan untuk mengetahui hubungan antara penelitian yang akan dilakukan dengan penelitian terdahulu, sehingga dengan menambahkan referensi tersebut penelitian yang akan dilakukan dapat menghindari adanya duplikasi. Berikut beberapa ulasan dari penelitian terdahulu yang berkaitan dengan penelitian yang dilakukan.

Penelitian pertama yang akan dibahas adalah penelitian yang dilakukan oleh Renjira Naufhal Dhiaegana (2020) dengan judul “*Penerapan Convolutional Neural Network Untuk Deteksi Pedestrian pada Sistem Autonomous Vehicle*”. Pada penelitian ini dibuat sebuah sistem yang mendeteksi *pedestrian* menggunakan Model *YOLO* versi empat serta *YOLO* versi empat *tiny*, yang telah dilatih dengan dataset *CrowdHuman*. Dari hasil pengujian performa model yang paling optimal adalah *YOLOv4-tiny-416-double* dengan nilai mAP 69,02.

Penelitian kedua adalah penelitian yang dilakukan oleh Muhammad Satrio Wicaksono (2020) dengan judul “*Pendeteksian dan Pelacakan Objek pada Lalu Lintas Padat dan Beragam untuk Mobil Otonom*”. Pada penelitian ini penulis membuat sebuah sistem persepsi pada mobil otonom dengan menggabungkan dua buah proses, yaitu pendeteksian objek dan pelacakan objek. Pada penelitian ini algoritma pendeteksian objek *YOLOv3* dibandingkan dengan *YOLO-SPP* yang merupakan *YOLOv3* dengan penambahan *layer SPP*. Dan algoritma *SORT* digunakan sebagai algoritma pelacakan objek. Dari hasil pengujian diperoleh bahwa model terbaik adalah *YOLO-SPP* yang dilatih dengan *transfer learning* dapat mencapai nilai mAP sebesar 95,49%. Hasil pengujian dari pelacakan dengan algoritma *SORT* dengan model *YOLO-SPP* diperoleh rata-rata nilai *MOTA* pada 87,73% dan rata-rata *MOTP* pada 95,03%.

Penelitian ketiga adalah penelitian yang dilakukan oleh Sisco Jupiyandi dkk. (2019) dengan judul “*Pengembangan Deteksi Citra Mobil untuk Mengetahui Jumlah Tempat Parkir Menggunakan CUDA dan Modified YOLO*”. Pada penelitian ini digunakan model *Modified YOLO* (*M-YOLO*) dengan bantuan komputasi

CUDA, untuk membangun sebuah sistem yang dapat mengetahui jumlah slot pada lahan parkir. Hasil dari penelitian ini menunjukkan bahwa dengan menggunakan GPU dibandingkan dengan CPU dapat mempercepat waktu komputasi rata-rata sebesar 0,179 detik dengan rata-rata akurasi sebesar 100%.

Tabel 2.1. Penelitian Terkait

| Peneliti | Judul | Objek | Tahun | Metode | Hasil |
|---------------------------|--|--|--------------|------------------------------|---|
| Geoffrey French, dkk. | Convolutional Neural Networks for Counting Fish in Fisheries Surveillance Video | <i>Ikan</i> | 2015 | Convolutional Neural Network | Rata- rata akurasi sebesar 90.94% |
| A. Sanchez-Ortiz, dkk. | Mosquito larva classification method based on convolutional neural networks | Telur Nyamuk | 2017 | Convolutional Neural Network | Akurasi tertinggi pada telur nyamuk 96,88% |
| Sisco Jupiyandi, dkk. | <i>Pengembangan Deteksi Citra Mobil untuk Mengetahui Jumlah Tempat Parkir Menggunakan CUDA dan Modified YOLO</i> | Mobil | 2019 | Modified YOLO | Rata-rata akurasi sebesar 100 dan Rata-rata waktu komputasi sebesar 0,179 detik |
| Muhammad Satrio Wicaksono | <i>Pendeteksian dan Pelacakan Objek pada Lalu Lintas Padat dan Beragam untuk Mobil Otonom</i> | <i>Bus, Car, Mini Pickup, Mini Truck Box, Minibus, Motorcyclist, Pedestrian, dan Truck</i> | 2020 | YOLO, YOLO-SPP dan SORT | Hasil terbaik adalah YOLO-SPP dengan skor 93,18% |

| Peneliti | Judul | Objek | Tahun | Metode | Hasil |
|------------------------------|---|-------------------|--------------|---------------------------|---|
| Renjira Naufhal Dhiaegana | <i>Penerapan Convolutional Neural Network Untuk Deteksi Pedestrian pada Sistem Autonomous Vehicle</i> | <i>Pedestrian</i> | 2020 | YOLOv4 dan YOLOv4-Tiny | Model paling optimal adalah YOLOv4 Tiny Double dengan skor mAP 69,02% |