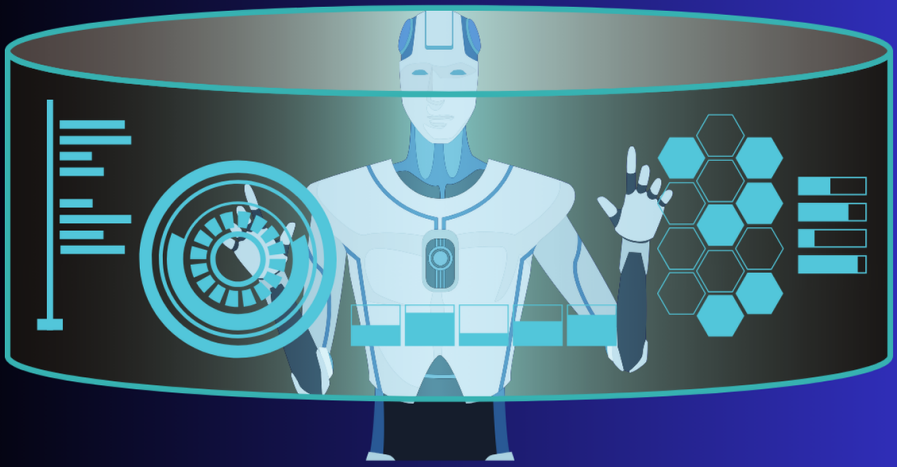
HEI PUBLISHING
CAKRAWALA JADI TULISAN

KUASAI MACHINE LEARNING & COMPUTER VISION DALAM SEKEJAP



Nandang Gunawan Tunggal Waras, S.Si, M.T.
Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM.
Ayunda Kusuma Wardani, S.Tr.Kom
Victor Benny Alexsius Pardosi, S.Kom., M.Sc.
Qonitatul Hasanah, S.S.T., M.Tr.T.
Arvita Agus Kurniasari, S.ST.,M.Tr.Kom.
Muhammad Hafidh Firmansyah, S.Tr.Kom., M.Sc.
Aji Seto Arifianto, S.ST, M.T
Giandari Maulani, S.Kom, M.Kom
Johar Nur Iin, S. Kom., MT

KUASAI MACHINE LEARNING & COMPUTER VISION DALAM SEKEJAP

Nandang Gunawan Tunggal Waras, S.Si, M.T.
Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM.
Ayunda Kusuma Wardani, S.Tr.Kom
Victor Benny Alexsius Pardosi, S.Kom., M.Sc.
Qonitatul Hasanah, S.S.T., M.Tr.T.
Arvita Agus Kurniasari, S.ST.,M.Tr.Kom.
Muhammad Hafidh Firmansyah, S.Tr.Kom., M.Sc.
Aji Seto Arifianto, S.ST, M.T
Giandari Maulani, S.Kom, M.Kom
Johar Nur Iin, S. Kom., MT

HEI PUBLISHING
CAKRAWALA JADI TULISAN

CV HEI PUBLISHING INDONESIA

KUASAI MACHINE LEARNING & COMPUTER VISION DALAM SEKEJAP

Penulis :
Nandang Gunawan Tunggal Waras, S.Si, M.T.
Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM.
Ayunda Kusuma Wardani, S.Tr.Kom
Victor Benny Alexsius Pardosi, S.Kom., M.Sc.
Qonitatul Hasanah, S.S.T., M.Tr.T.
Arvita Agus Kurniasari, S.ST.,M.Tr.Kom.
Muhammad Hafidh Firmansyah, S.Tr.Kom., M.Sc.
Aji Seto Arifianto, S.ST, M.T
Giandari Maulani, S.Kom, M.Kom
Johar Nur Iin, S. Kom., MT

ISBN : 978-623-8722-12-9
Editor : Ade Wisandra, S.Kom, M.Kom
Penyunting : Muhammad Ikhlas Al Kutsi, S.Kom., S.Pd., M.M.
Desain Sampul dan Tata Letak : Ipah Kurnia Putri S.ST

Penerbit : CV HEI PUBLISHING INDONESIA
Anggota IKAPI No. 034/SBA/2023

Redaksi :
Jl. Air Paku No.29 RSUD Rasidin, Kel. Sungai Sapih, Kec Kuranji
Kota Padang Sumatera Barat
Website : www.HeiPublishing.id
Email : heipublishing.id@gmail.com

Cetakan pertama, Juli 2024

KATA PENGANTAR

Dengan mengucapkan puji syukur kehadirat Allah SWT, atas limpahan rahmat dan hidayahNya, maka Penulisan Buku dengan judul **Kuasai Machine Learning & Computer Vision dalam Sekejap** dapat diselesaikan. Buku ini berisikan bahasan tentang Compuer Vision, Machine Learning, Pemrogaman Python, Persiapan dan Pra Pemrosesan Data, Computer Vision Dengan Conlutional Neural Networks, Computer Vision Dengan YOLO, Computer Vision Dengan Regression dan Computer Vision Dengan Clustering

Buku ini masih banyak kekurangan dalam penyusunannya. Oleh karena itu, kami sangat mengaharapkan kritik dan saran demi perbaikan dan kesempurnaan buku ini selanjutnya. Kami mengucapkan terima kasih kepada berbagai pihak yang telah membantu dalam proses penyelesaian Buku ini. Semoga Buku ini dapat menjadi sumber referensi dan literatur yang mudah dipahami.

Padang, Juli 2024
Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

Oleh Nandang Gunawan Tunggal Waras, S.Si, M.T.

1. Ikhtisar

Pembelajaran Mesin (*Machine Learning)* adalah bidang yang memusatkan mulai dari perancangan dan analisis algoritma sehingga komputer untuk belajar (Ibnu Daqiqil ID, 2021). Menurut Arthur Samuel, Pembelajaran Mesin di dalamnya terdapat algoritma yang bersifat umum dan algoritmanya dapat mengeluarkan keluaran yang bermanfaat. Poin utamanya adalah algoritma tersebut saat ditambahkan sejumlah data maka dapat terbangun sebuah aturan/model/inferensi dari data tersebut.

Pembelajaran Mesin juga tentang mengekstraksi pengetahuan dari data (Budi Raharjo, 2021). Pembelajaran Mesin adalah bidang penelitian di yang mengombinasikan bidang ilmu komputer, statistik dan kecerdasan buatan, dan biasa disebut analitik prediktif/pembelajaran statistik. Misalnya, terdapat usulan otomatis tentang tontonan, hingga makanan yang dapat diorder atau barang yang akan dibeli.

Di luar aplikasi berbayar, pembelajaran mesin memiliki efek yang luar biasa pada penelitian yang menggunakan data. *Tools* yang telah diterapkan pada beragam masalah ilmiah seperti memahami benda-benda langit, menemukan partikel baru dan lain sebagainya.

Pemanfaatan aplikasi pembelajaran mesin sudah sangat terbiasa dalam kehidupan keseharian. Aktivitas-

aktivitas yang seperti sistem rekomendasi, peramalan, diagnosis, dan lain sebagainya. Contohnya adalah buatan Google, misalnya Google Terjemahan (terjemahan mesin, pengenalan tulisan tangan, pengenalan suara, Alpha Go), Google Search dan Gmail (Ibnu Daqiqil ID, 2021).

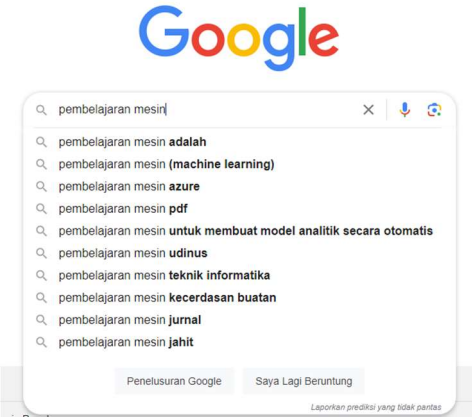
Gambar 1. Hal-hal yang dapat dibantu penyelesaiannya dengan pembelajaran mesin (Sumber : Medium.com)

Di bawah ini terdapat contoh implementasi pembelajaran mesin dalam kehidupan sehari-hari (Ibnu Daqiqil ID, 2021):

1. Mesin Pencari

Mesin-mesin pencari seperti Google, Bing dan Yandex yang mengimplementasikan pembelajaran mesin untuk melakukan pemeringkatan laman suatu situs, rekomendasi kata kunci, perbaikan jika ada pengejaan dan rekomendasikata kunci yang lebih baik. Setiap mesin

pencari memiliki "ramuan" masing-masing yang terdapat pada algoritma pencariannya.



Gambar 2. Google memberikan rekomendasi kata kunci pencarian

Dalam implementasinya saat kata kunci diketikkan, Google akan menampilkan hasil pencarian yang paling mendekati**.** Apabila dipilih suatu halaman dan mengalokasikan waktu yang lebih banyak pada halaman tersebut**,** Google akan mengenali bahwa halaman tersebut memiliki kesesuaian dengan kata kunci yang diketikkan. Google akan mengenali adanya ketidaksesuaian kata kunci dengan hasil pencarian yang dihasilkan. Dan data tersebut secara otomatis terkumpul dan diakumulasikan menggunakan pembelajaran mesin oleh Google Search Engine**.** Hal ini dimaksudkan untuk menghasilkan hasil pencarian yang lebih dinamis dan bermutu.

2. *Online Shop* dan *Marketplace*

Penggunaan pembelajaran mesin di *marketplace* atau toko Online dapat memberikan profit baik bagi penjual

maupun pembeli. Agar pengalaman pengguna dan meningkatkan penjualan lebih meningkat, setiap akun perlu menampilkan rekomendasi barang yang sesuai dengan minat pembeli. Untuk melakukan ini secara otomatis dan *real-time,* tentunya pembelajaran mesin sangat menentukan akurasi rekomendasi produk tiap pembeli di akunnya.

3. Media Sosial

Media Sosial adalah program yang tidak dapat lepas dalam aktivitas keseharian masyarakat kita. Karena itu penyedia layanan selalu meningkatkan fitur dari aplikasi agar penggunanya merasa mudah dan aman. Contohnya instagram, ia menggunakan beberapa teknologi pembelajaran mesin untuk memberikan rekomendasi konten iklan bagi penggunanya.

4. Iklan Digital

Google Adword menampilkan iklan pada website. Iklan yang muncul adalah iklan yang berubah-ubah. Google Adword mengumpulkan dan mengelompokkan data situs berdasarkan topiknya, kemudian menampilkan iklan yang berhubungan dengan topik tersebut. Selain itu, Google Adword juga menggunakan *cookies*, sebagai referensi aktivitas pengunjung suatu website terkait website yang dikunjungi sebelumnya.

5. Asisten Pribadi Virtual

Telepon pintar, tablet dan komputer jinjing, saat ini telah dilengkapi dengan asisten pribadi virtual, seperti : Google Now di Android, Cortana di Microsoft Windows, Siri di Iphone. Asisten ini dapat membantu penggunanya untuk melakukan pencarian di internet, memberikan informasi

tentang kondisi jalan, cuaca saat ini dan akan datang, panggilan telepon, dan lain-lain.

6. Penyaring Pesan Spam

Pembelajaran mesin memiliki peran yang sangat besar untuk melakukan penyaringan spam baik di surat elktronik, laman web, hingga media komunikasi yang sudah terpasang. Algoritma pohon keputusan merupakan cikal bakal dari algoritma *filtering,* untuk mennyaring suatu pesan termasuk spam atau bukan.

7. Mobil Kendali Otomatis

Mobil kendali otomatis adalah implementasi dan pengembangan dari pembelajaran mesin yaitu *machine vision*. Mobil kendali otomatis merupakan penetapan pembelajaran mesin yang kompleks dan resiko langsung yang tinggi (*highrisk*). Mobil harus mempelajari banyak hal, mulai dari pengenalan rambu lalu lintas, arah dan tujuan, kondisi jalan, lampu lalu lintas*,* situasi manusia di sekitarnya, dan sensor-sensor yang terintegrasi di dalamnya.

2. Machine Learning dalam Computer Vision

Visi Komputer adalah disiplin ilmu dan teknologi yang bisa menginterpretasikan informasi dari gambar untuk mengetahui informasi dan menyelesaikan tugas tertentu. Teori ini merupakan salah satu sistem kercerdasan buatan yang ekstrak informasi dari gambar. Data gambar tersebut bisa berupa banyak bentuk, urutan video, sudut pandangan dari berbagai kamera, atau scanning hasil gambar. Cabang ini merupakan bagian dari Artificial Intellegence yang mengenali obyek yang diamati dan mampu menghasilkan visual intellegence system yang dapat mengenali obyek yang diamati. Jadi, bagaimana komputer

dapat mengenali obyek yang akan diamati (Giandari Maulani dkk, 2024).

Computer Vision adalah kombinasi dari Image Processing atau penglolahan citra yang berhubungan dengan transformasi citra/gambar yang memiliki tujuan untuk menghasilkan mutu yang lebih baik dan kombinasi dari pengenalan pola yang bertujuan dalam proses identifikasi obyek pada interpretasi citra untuk mengekstrak informasi/pesan yang disampaikan citra/gambar. Contoh sederhana penerapan komputer visi adalah identifikasi wajah orang tertentu atau identifikasi sidik jari dari anggota badan seseorang atau identifikasi sebuah barang atau alat transportasi tertentu.

Visi komputer juga merupakan dan kumpulan dari metode untuk memperoleh, memroses, menganalisis suatu gambar. Dalam arti lain, visi komputer merupakan kumpulan metode yang digunakan untuk menghasilkan angka atau simbol yang diperoleh dari gambar yang diambil dari dunia nyata supaya komputer dapat mengerti apa arti gambar tersebut. Inti dari teknologi visi komputer adalah untuk menduplikasi kemampuan penglihatan manusia ke dalam benda elektronik sehingga benda elektronik dapat memahami dan mengerti arti dari gambar yang dimasukkan.

Visi komputer juga dideskripsikan sebagai suatu teknologi untuk mengotomatisasi dan mengintegrasikan berbagai proses dan representasi untuk menghasilkan presepsi penglihatan (Dana H. Ballard dan Christopher M. Brown, 1982).

Dalam disiplin ilmu, visi komputer berkaitan dengan teori-teori dibalik sistem buatan yang mengekstrak informasi dari gambar. Informasi yang diekstrak dari gambar dapat berupa data-data yang berbeda-beda, seperti urutan

jalannnya video, intesitas cahaya, atau prespektif dari sudut gambar yang berbeda beda. Dalam disiplin teknologi, visi komputer mengusahakan cara agar teori-teori dan model dapat diterapkan untuk pembangunan sistem pada sistem komputer. Contoh aplikasi yang menggunakan visi komputer seperti, alat navigasi dan kontroler.

Saat ini Visi komputer kerapkali digunakan untuk mendeteksi wajah pada gambar mengenali ekspresi wajah dan prakteknya sering diaplikasikan bersama dengan jaringan syaraf tiruan.

Pembelajaran mesin dan visi komputer diharapkan dapat menghadirkan kemampuan manusia dalam penginderaan data, pemahaman data, dan pengambilan tindakan ke dalam komputer berdasarkan hasil masa lalu dan masa kini (Asharul Islam Khan dan Salim al Habsi, 2019). Penelitian pembelajaran mesin dan visi komputer masih terus berkembang. Visi komputer adalah bagian penting dari *Internet of Things*, *Industrial Internet of Things*, dan antarmuka otak manusia. Aktivitas manusia yang kompleks dikenali dan dipantau dalam aliran multimedia menggunakan pembelajaran mesin dan visi komputer. Metode ini menggunakan algoritme pembelajaran mesin seperti mesin vektor dukungan, CNN, dan lain-lain. Solusi pembelajaran mesin berkisar pada pengumpulan data, melatih model, dan menggunakan model terlatih untuk membuat prediksi. Ada model dan layanan yang disediakan oleh perusahaan swasta untuk pengenalan suara, analisis teks, dan klasifikasi gambar. Seseorang dapat menggunakan modelnya melalui antarmuka pemrograman aplikasi (API). Misalnya, Amazon Recognition, Polly, Lex, Microsoft Azure Cognitive Services, IBM Watson. teksi dan analisis objek adalah bagian penting dalam kehidupan sehari-hari. Deteksi objek memiliki aplikasi dalam

menghindari tabrakan lalu lintas, pengenalan ekspresi wajah dan pengenalan emosi berdasarkan postur tubuh manusia. Pada dikembangkan sistem otomatis untuk mendeteksi informasi yang terdapat pada wajah manusia dari gambar dan video dengan bantuan orientasi. TensorFlow dan OpenPose adalah perpustakaan perangkat lunak yang digunakan dalam deteksi objek dan visi komputer. Model deteksi lalu lintas menggunakan jaringan saraf convolutional, jaringan saraf berulang (RNN), memori jangka pendek panjang (LSTM), gated recurrent unit (GRU), dan jaringan Bayesian. Dalam lingkungan cerdas, sensor menangkap data yang kemudian digunakan untuk analisis dan peramalan Ekstraksi fitur adalah salah satu tugas jaringan saraf konvolusi (CNN) yang diselesaikan tanpa kehilangan informasi agar deteksi objek berhasil. Pembelajaran yang diawasi dari jaringan saraf konvolusional dalam mengenali wajah dengan seju Pembelajaran Mesinah besar gambar wajah. Satu-satunya tantangan dalam penerapan visi komputer dan pembelajaran mesin adalah anotasi/pelabelan data. Algoritma pembelajaran mesin sekarang berjalan di cloud sebagai "pembelajaran mesin sebagai layanan", "pembelajaran mesin cloud" . Selain itu, perusahaan seperti Amazon, Microsoft, dan Google, memiliki pembelajaran mesin sebagai layanan cloud.

Ada tiga pendekatan untuk pembelajaran mesin dan visi komputer: pembelajaran yang diawasi, tanpa pengawasan, dan semisupervisi. Pembelajaran yang diawasi telah memberi label pada data pelatihan. Pelabelan data mahal, memakan waktu dan memerlukan keahlian. Di sisi lain, pembelajaran semi-supervisi memiliki sebagian data yang diberi label dan sebagian lainnya tidak. Pengklasifikasi jaringan Bayesian memiliki keunggulan dalam pembelajaran dengan data yang tidak berlabel. Namun demikian, permasalahan dunia nyata termasuk dalam kategori pembelajaran tanpa

pengawasan di mana pola berkembang berdasarkan pengelompokan. Paradigma pembelajaran mesin untuk visi komputer adalah mesin vektor pendukung, jaringan saraf, dan model grafis probabilistik.

Support vector machine (SVMs) adalah subdomain dari metode pembelajaran mesin yang diawasi dan populer dalam klasifikasi. Jaringan saraf terdiri dari jaringan berlapis dari node pemrosesan yang saling berhubungan. Jaringan saraf konvolusional (CNN) adalah kategori jaringan saraf yang digunakan dalam pengenalan dan klasifikasi gambar. Ia memiliki neuron dengan dimensi: lebar, tinggi dan kedalaman. CNN telah mendapatkan popularitas akhir-akhir ini karena sebagian besar kumpulan data, GPU, dan teknik regularisasi yang dapat diakses. OpenCV adalah perpustakaan (*library*) yang dapat diintegrasikan dengan bahasa pemrograman seperti Android, .NET, Java, iOS pada platform seperti Eclipse dan Visual Studio di Windows, iOS, dan Linux untuk pemrosesan dan analisis gambar. Ini digunakan dalam pemrosesan gambar, analisis video, deteksi objek, dan pembelajaran mesin. Gambar 1 menunjukkan proses deteksi objek di lingkungan pembelajaran mesin dan visi komputer.

Gambar 3. Proses deteksi objek di lingkungan pembelajaran mesin dan visi komputer. (Aslam Islam Khan dan Salim Al Habsi, 2019)

Studi ini mengeksplorasi berbagai aplikasi pembelajaran mesin dalam visi komputer. Misalnya saja

segmentasi,ekstraksi fitur, penyempurnaan model visual, pencocokan pola, representasi bentuk, rekonstruksi permukaan, dan pemodelan untuk ilmu biologi. Pembelajaran mesin dalam computer vision digunakan dalam menafsirkan data yang ada di dalam mobil dan gambar deteksi pejalan kaki, klasifikasi otomatis kesalahan pada jalur kereta api menggunakan gambar, interpretasi data penginderaan jauh untuk sistem informasi geografis, membedakan veritas mangga berdasarkan pada atribut ukuran, ekstraksi informasi grafis dan tekstual dari gambar dokumen. Demikian pula lainnya aplikasi termasuk pengenalan gerakan dan wajah , visi mesin, mengenali karakter tulisan tangan dan digit, sistem bantuan pengemudi tingkat lanjut, studi perilaku, dan estimasi seluruh tubuh manusia kinematika untuk pengendara sepeda dan estimasi pose .

Dalam, mempelajari penggunaan computer vision dan pembelajaran mesin dalam ilmu kedokteran seperti pencitraan kardiovaskular, pembuluh darah retina, kedokteran nuklir,endoskopi, termografi, angiografi, resonansi magnetik, USG dan mikroskop. Pembelajaran mesin dan visi komputer memiliki aplikasi inovatif dalam bidang teknik, obat-obatan, pertanian, astronomi, olahraga, pendidikan, dan lain-lain.

3. Tujuan dan Ruang Lingkup Buku

Tujuan penyusunan buku ini adalah untuk memberikan pengetahuan dan wawasan mengenai peran pembelajaran mesin (*machine learning*) dalam implementasinya dalam kehidupan sehari-hari.

Ada pun ruang lingkup buku ini terdiri atas 10 (sepuluh) Bab, yang terdiri dari Bab 1 Pendahuluan, Bab 2 tentang Computer Vision yang mencakup definisi computer vision, Pengolahan Citra Digital dan Ekstraksi Fitur. Bab 3 tentang

Machine Learning yang mencakup konsep dasar machine learning, supervised dan unsupervised. Bab 4 tentang pemrograman python. Bab 5 tentang Persiapan dan prapemrosesan data. Bab 6 hingga Bab 10 mencakup studi kasus machine learning yang menggunakan *Convulational Neural Network* (CNN), *Single Shot Multibox Detector* (SSD), *You Only Look Once* (YOLO), *Registering* dan *Clustering* dengan bahasa pemrograman python.

DAFTAR PUSTAKA

Daqiqil, Ibnu. 2021, Machine Learning: Teori, Studi Kasus dan Implementasi Menggunakan Python. Penerbit UR Press. Riau.

Raharjo, Budi. 2021 Pembelajaran Mesin (Machine Learning).Yayasan Prima Agus Teknik.Semarang.

Dana Harry Ballard , Christopher M.Brown, 1982. Computer Vision. Prentice Hall.

Maulani, Giandari dkk. 2023. Development of Artificial Intelegence Application.Hei Publishing Indonesia.Padang

Asharul Islam Khan dan Salim Al-Habsi. 2019. Machine Learning in Computer Vision. International Conference on Computational Intelligence and Data Science (ICCIDS 2019)

BAB 2
COMPUTER VISION

Oleh Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM.

1. Apa yang Dimaksud dengan Computer Vision

Computer Vision berarti studi tentang ilmu visi yang memungkinkan desain perangkat lunak pada tingkat yang lebih rendah dengan apa yang masuk ke dalam sistem visi terintegrasi. *Machine Vision* mempunyai studi yang tidak hanya berfokus mengenai perangkat lunak tetapi berhubungan dengan lingkungan perangkat keras dan teknik akuisisi gambar yang diperlukan untuk suatu aplikasi nyata sehingga berorientasi pada teknik (Davies, 2012).

Computer Vision (CV) membawa manusia kepada dunia yang menarik di mana komputer dapat memahami, menganalisis, dan menginterpretasi gambar dan video secara mirip dengan cara manusia (Dompeipen and Sompie, 2020). *Computer Vision* merupakan subbidang ranah dalam kecerdasan buatan (*artificial intelligence*) dengan tujuan untuk mengotomatisasi aktivitas yang berkaitan pengembangan teknologi memproses dan menganalisis visual informasi dalam bentuk *digital* (Wahyudi, 2023).

Sistem *computer vision* dibangun melalui perangkat keras (*hardware*) berupa kamera, komputer dan perangkat pendukung yang lain. Sistem memerlukan perangkat lunak (*software*) untuk menciptakan *script* (perintah) melalui pengolahan citra gambar. Perangkat lunak yang dibutuhkan sebagai alat bantu pada aplikasi pemrograman akan

digunakan dalam upaya memproses citra gambar seperti mengambil (*take an image*), merubah atau konversi (*change or convert images*), membaca data piksel (*read pixel data*), komputasi data piksel (*pixel data computing*), dan klasifikasi hasil perhitungan yang menyediakan keputusan (Subur *et al.*, 2024).

Teknologi berkembang sangat pesat sehingga membuat *computer vision* telah diterapkan, difungsikan dan dimanfaatkan dalam aplikasi modern di berbagai ranah bidang, seperti:

1. Pengenalan gambar: kemampuan mengenali objek, wajah, atau pola dalam gambar.
2. Pengolahan citra medis: kemampuan mendeteksi penyakit atau abnormalitas dalam gambar medis seperti *MRI* atau *CT scan*.
3. Penglihatan komputer untuk kendaraan otonom: digunakan dalam kendaraan otonom untuk mendeteksi rambu lalu lintas, pejalan kaki, atau kendaraan lain di sekitar.
4. Keamanan dan pengawasan: digunakan untuk pengawasan kamera, deteksi gerakan, dan pengenalan wajah dalam berbagai penerapan sistem keamanan.
5. Pengenalan tulisan tangan dan karakter: kemampuan dalam mengenali dan menginterpretasi tulisan tangan manusia atau karakter dalam dokumen cetak maupun *digital*.
6. Realitas augmentasi: digunakan dalam berbagai aplikasi realitas augmentasi (AR) dan realitas *virtual* (VR) untuk menggabungkan elemen *visual* dengan dunia nyata atau menciptakan lingkungan *virtual* yang relevan.
7. Pengawasan lingkungan: digunakan untuk berbagai pemantauan di sekitar lingkungan strategis, seperti deteksi kebakaran hutan atau pengamatan aliran lalu lintas jalan raya.

Computer Vision telah membuka pintu untuk berbagai aplikasi yang menguntungkan dan strategis di berbagai bidang layanan. Kemampuan dalam memahami dan menganalisis gambar dan *video* membuat *computer vision* membuka peluang baru dalam berbagai industri dan aplikasi, menghasilkan inovasi yang signifikan dalam cara bekerja, hidup, dan berinteraksi dengan dunia di sekitar.

Gambar 4. Computer Vision

Perkembangan Ilmu Pengetahuan, Teknologi, dan Seni (IPTEKS) menghasilkan perangkat yang akan membantu manusia dalam menyelesaikan pekerjaan secara otomatis (Manakitsa *et al.*, 2024). *Computer Vision* akan melibatkan konsep dasar kunci dalam membentuk berbagai pemahaman nyata mengenai bagaimana komputer memproses serta dapat memahami gambar dan *video* (Prabowo, Abdullah and Ari Manik, 2018). Konsep dasar dalam *computer vision* memiliki ketentuan sebagai berikut:

a. Representasi citra: merupakan cara gambar direpresentasikan dalam komputer yang mencakup format seperti citra *grayscale*, citra berwarna (*RGB*), dan citra

dalam format lain seperti *HSV* (*hue*, *saturation*, *value*). Setiap piksel dalam citra memiliki nilai tertentu yang mewakili warna atau intensitas, dan komputer akan memproses citra berdasarkan nilai tersebut.

b. Praproses citra: melibatkan berbagai rangkaian teknik untuk mempersiapkan citra sebelum diproses lebih lanjut seperti normalisasi intensitas, pengurangan *noise*, dan penghalusan citra untuk meningkatkan kualitas gambar dan memudahkan analisis.
c. Segmentasi citra: merupakan proses membagi citra menjadi bagian yang berbeda, seperti objek dan latar belakang yang bertujuan mengidentifikasi dan memisahkan objek dari latar belakang yang memungkinkan analisis pada objek tersebut.
d. Ekstraksi fitur: melibatkan identifikasi fitur penting dalam citra yang dapat digunakan untuk tujuan tertentu seperti deteksi objek atau pengenalan pola dengan fitur berupa tepi, sudut, dan atau tekstur dalam citra.
e. Pengenalan objek: merupakan proses identifikasi dan klasifikasi objek citra yang menggunakan algoritma teknik pengenalan pola (*machine learning*) untuk membandingkan fitur citra dengan model yang telah dipelajari untuk mengenali objek tertentu.
f. Pelacakan objek: merupakan proses melacak pergerakan objek dalam serangkaian gambar atau *video* dengan tujuan untuk mengidentifikasi dan melacak objek dari satu *frame* ke *frame* berikut sehingga memungkinkan analisis berjalan secara dinamis melalui objek dalam konteks temporal.
g. Klasifikasi citra: merupakan proses mengelompokkan citra ke dalam kategori tertentu berdasarkan karakteristik atau fitur yang diekstraksi yang meliputi klasifikasi objek, klasifikasi aksi, atau klasifikasi konten visual lain.

h. Rekonstruksi 3D: melibatkan pembuatan model tiga dimensi dari dunia nyata yang ditentukan berdasarkan nilai informasi citra.

Konsep dasar dalam pengembangan *computer vision* sangat penting sebagai fondasi utama untuk mengetahui, mengidentifikasi, dan mengembangkan sistem yang mampu memahami cara bekerja dengan gambar dan *video*.

2. Teknik Pengolahan Citra Digital

Citra merupakan kombinasi antara titik, garis, bidang, dan warna yang membuat suatu imitasi dalam bentuk objek fisik atau manusia (Hendryanto Edha, Sampe Hotlan Sitorus, 2020). Citra adalah suatu representasi, kemiripan, atau imitasi dari objek yang bersifat *analog* dan *digital*. Citra *analog* merupakan citra yang memiliki karakter *continue* seperti gambar pada monitor, televisi, foto sinar X, dan yang lain. Citra *digital* merupakan citra yang dikelola melalui perangkat komputer. Citra berarti sebagai fungsi f(x,y) berukuran M baris dan N kolom, di mana x dan y adalah koordinat spasial, dan amplitudo f di titik koordinat (x,y) dengan intensitas atau tingkat keabuan dari citra (F., Yuwono and P. Boedi, 2022).

Teknik *Digital Image Processing* (Pengolahan Citra Digital) merupakan ranah ilmu yang berfokus pada manipulasi dan analisis gambar *digital* dengan menggunakan algoritma komputer untuk mengetahui proses citra akan dibentuk, diolah, dan dianalisis sehingga memperoleh informasi yang akan dipelajari manusia pengguna (Yuhandri, R. and Syahputra, 2022).

Pengolahan citra berguna meningkatkan kualitas gambar, mengekstrak informasi, melakukan seleksi citra (*feature images*), melaksanakan kompresi data (reduksi data), dan interpretasi. Peningkatan kualitas bertujuan untuk

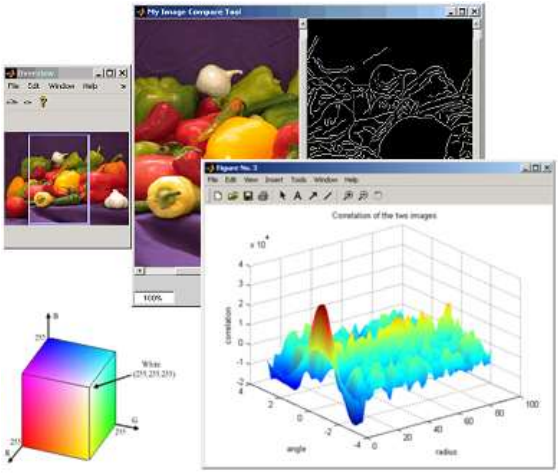
menyesuaikan bentuk citra agar memperoleh hasil yang relevan (Nabusa, 2019).

Teknik *Digital Image Processing* menggunakan berbagai metode, cara, dan algoritma komputer untuk dapat melakukan manipulasi dan analisis gambar *digital*. Secara umum teknik yang digunakan meliputi:

1. *Pre-processing* (pra-pemrosesan): proses pengolahan citra yang melibatkan operasi seperti *filtering*, *noise removal*, dan normalisasi untuk meningkatkan kualitas gambar.
2. *Enhancement* (peningkatan): teknik untuk meningkatkan kualitas gambar dengan meningkatkan kejelasan, kontras, atau detail dari gambar seperti *histogram equalization*, *contrast stretching*, dan *sharpening*.
3. *Segmentation* (segmentasi): proses membagi gambar menjadi beberapa bagian atau objek yang mempunyai karakteristik serupa sehingga dapat membantu dalam mengidentifikasi objek dalam gambar hingga menganalisis struktur gambar.
4. *Feature Extraction* (ekstraksi fitur): identifikasi dan ekstraksi fitur penting dari gambar yang digunakan untuk mewakili informasi penting dalam citra. Fitur berupa tekstur, warna, bentuk, atau pola.
5. *Recognition* (pengenalan): menggunakan fitur yang diekstraksi untuk mengenali atau mengklasifikasikan objek dalam gambar sehingga melibatkan penggunaan metode seperti klasifikasi berbasis *machine learning* atau pengenalan pola.
6. *Restoration* (pemulihan): teknik untuk memulihkan gambar yang telah rusak atau terganggu dengan teknik seperti pemulihan citra berbasis model, *deblurring*, dan *denoising*.

7. *Compression* (kompresi): teknik untuk mengurangi ukuran gambar tanpa kehilangan terlalu banyak informasi yang dilakukan dengan metode kompresi *lossy* dan *lossless*.
8. *Image Analysis* (analisis citra): penerapan algoritma dan teknik statistik untuk menganalisis gambar dalam konteks aplikasi tertentu, seperti pengolahan citra medis, pemrosesan citra satelit, atau pengenalan wajah.

Pemilihan teknik yang relevan tergantung pada tujuan spesifik dari analisis citra yang dilakukan dan setiap teknik mempunyai aplikasi tertentu seperti kedokteran, robotika, dan pemrosesan citra satelit.



Gambar 5. Pengolahan Citra *Digital*

3. Ekstraksi Fitur (Warna, Bentuk, Tekstur)

Cara kerja *computer vision* melibatkan serangkaian langkah dan teknik untuk memproses informasi *visual* di mana berupaya meniru sistem kerja *visual* manusia (*human vision*) yang rumit atau kompleks (Komputer and Segmentasi, 2010).

Ekstraksi fitur adalah proses memperoleh fitur dari citra *digital* yang memanfaatkan metode tertentu untuk mengkalkulasi jumlah piksel pada citra (Hidayat *et al*., 2021). Ekstraksi Fitur (Fitur Citra) merupakan proses mengidentifikasi dan mengekstraksi informasi penting dari gambar yang digunakan untuk mewakili karakteristik yang berguna (Diantarakita and Rahman, 2019). Jenis fitur citra yang diekstraksi meliputi warna, bentuk, dan tekstur.

1. Fitur Warna

   Fitur warna mencakup distribusi warna dalam gambar yang melibatkan ekstraksi statistik sederhana seperti rerata dan deviasi standar warna, atau dapat menggunakan representasi warna yang lebih kompleks seperti model warna *RGB*, *HSV*, atau *LAB*. Ekstraksi fitur warna banyak digunakan dalam aplikasi pengenalan objek, klasifikasi gambar, dan segmentasi warna.

   Fitur warna dalam *computer vision* mengacu pada representasi dan analisis distribusi warna dalam gambar *digital*. Fitur warna dapat menjadi salah satu aspek penting dalam proses pengenalan objek, segmentasi, klasifikasi, dan pemrosesan citra yang lain.

   Beberapa aspek penting tentang fitur warna dalam penerapan *computer vision* meliputi:

   a. Representasi warna: merupakan gambar *digital* yang direpresentasikan dalam berbagai model warna, seperti *RGB* (*Red, Green, Blue*), *HSV* (*Hue, Saturation, Value*), *LAB*, *YUV*, dan yang lain. Setiap model mempunyai keunggulan dan kekurangan tersendiri yang tergantung pada aplikasi dan jenis analisis yang dilakukan.
   b. *Histogram* warna: merupakan representasi distribusi frekuensi intensitas warna dalam gambar yang dapat memberikan informasi tentang seberapa sering setiap nilai warna muncul dalam gambar. *Histogram* warna

digunakan untuk analisis warna dan segmentasi warna dalam gambar.

c. Deteksi warna: merupakan penggunaan fitur warna yang dapat membantu dalam deteksi objek berdasarkan warna seperti dalam deteksi bola pada lapangan sepak bola di mana fitur warna dapat digunakan untuk menemukan berbagai daerah dengan warna merah atau kuning yang sesuai dengan jenis dan bentuk bola.
d. Segmentasi warna: dimanfaatkan pada segmentasi citra yang memisahkan objek atau area dengan warna tertentu dari latar belakang. Metode seperti *thresholding* warna atau pemodelan warna berbasis *cluster* digunakan untuk mencapai tujuan.
e. Ekstraksi fitur: statistik memiliki fitur rerata warna, deviasi standar warna, atau momen warna yang dapat diekstraksi dari gambar untuk digunakan dalam analisis lebih lanjut atau dalam proses klasifikasi.
f. Pengenalan warna: penggunaan fitur warna akan membantu dalam pengenalan objek berdasarkan warna yang digunakan dalam aplikasi seperti pengenalan pola, deteksi kebakaran, atau sistem visi mesin untuk memisahkan objek berdasarkan warna yang dominan.

Fitur warna merupakan aspek yang penting dan strategis dalam analisis citra *digital* karena warna sering menjadi fitur *visual* yang kuat dan mudah dikenali oleh manusia. Penggunaan aplikasi *computer vision* akan memanfaatkan fitur warna yang dapat memberikan informasi sangat berharga untuk berbagai tugas analisis dan pengolahan citra yang dibutuhkan.

2. Fitur Bentuk

Fitur dasar dalam *visual content* pada citra di mana setiap objek citra dapat dibedakan melalui bentuk dari suatu objek tertentu. Fitur bentuk yang melibatkan ekstraksi informasi

geometris dari berbagai objek dalam gambar yang mencakup ekstraksi parameter bentuk seperti ukuran, bentuk, orientasi, hingga *convex hull*. Metode transformasi *Hough* digunakan untuk mengekstraksi garis dan bentuk geometris yang lain.

Ekstraksi fitur bentuk banyak digunakan dalam pengenalan objek dan pengenalan pola tertentu yang relevan. Fitur bentuk juga digunakan untuk menentukan area (wilayah) yang adalah jumlah piksel dalam lokasi digambarkan melalui suatu bentuk.

Fitur bentuk merupakan representasi geometris dari objek dalam gambar *digital*. Pengenalan dan ekstraksi fitur bentuk penting untuk bermacam aplikasi seperti pengenalan objek, deteksi objek, segmentasi, dan analisis citra. Berikut adalah beberapa aspek penting tentang fitur bentuk dalam *computer vision*:

a. Deskripsi geometris: fitur bentuk mencakup atribut geometris dari objek, seperti ukuran, bentuk, orientasi, dan struktur yang dapat diwakili dalam berbagai cara, termasuk koordinat titik-titik kontur objek, properti geometris seperti luas, perimeter, lingkaran minimum yang melingkari objek, atau deskriptor bentuk seperti *moments invariant*.
b. Transformasi *hough*: merupakan teknik yang dimanfaatkan dalam mendeteksi garis lurus, lingkaran, atau bentuk geometris lain dalam gambar sehingga berguna mengekstraksi fitur bentuk seperti garis atau sudut gambar.
c. Kontur: merupakan garis yang menggambarkan batas objek dalam gambar. Ekstraksi kontur dalam melakukan analisis dapat memberikan informasi tentang bentuk objek, serta memfasilitasi deteksi dan segmentasi objek.

d. Ekstraksi fitur morfologi: operasi morfologi seperti dilasi (morfologi), erosi, *opening*, dan *closing* yang digunakan untuk mengubah dan mengekstraksi fitur bentuk dari gambar. Dilasi dapat digunakan untuk mengisi lubang kecil dalam objek, sementara erosi dapat menghilangkan detail kecil.
e. Deskriptor *bounding box*: merupakan kotak pembatas yang mengelilingi objek dalam gambar. Ekstraksi fitur bentuk seperti koordinat titik sudut atau ukuran *bounding box* dapat memberikan informasi geometris tentang objek yang dapat digunakan dalam analisis lebih lanjut.
f. Pengenalan objek berbasis bentuk: fitur bentuk yang secara umum digunakan dalam pengenalan objek berbasis bentuk, di mana objek dikenali berdasarkan karakteristik geometris. Metode seperti *template matching* atau metode berbasis model geometris digunakan untuk mencocokkan objek dalam gambar dengan bentuk yang telah diketahui sebelumnya.

Fitur bentuk memainkan peran penting dalam analisis citra *digital* karena membawa informasi geometris yang kaya tentang objek dalam suatu gambar. Penggunaan fitur bentuk dapat mengidentifikasi, memisahkan, dan mengklasifikasikan objek dalam gambar dengan lebih baik sehingga merupakan langkah kunci dalam berbagai penggunaan aplikasi untuk memahami konsep *computer vision*.

3. Fitur Tekstur

Fitur tekstur menggambarkan pola spasial dari intensitas piksel dalam gambar yang mencakup karakteristik seperti keteraturan, kasar halus dan pola yang terlihat dalam citra. Metode ekstraksi fitur tekstur termasuk menggunakan metrik statistik seperti energi, kontras, dan korelasi, serta menggunakan deskriptor tekstur seperti *filter Gabor* atau

Gray Level Co-occurrence matrix (GLCM). Ekstraksi fitur tekstur berguna dalam aplikasi seperti deteksi tekstur, klasifikasi citra, dan pengenalan objek.

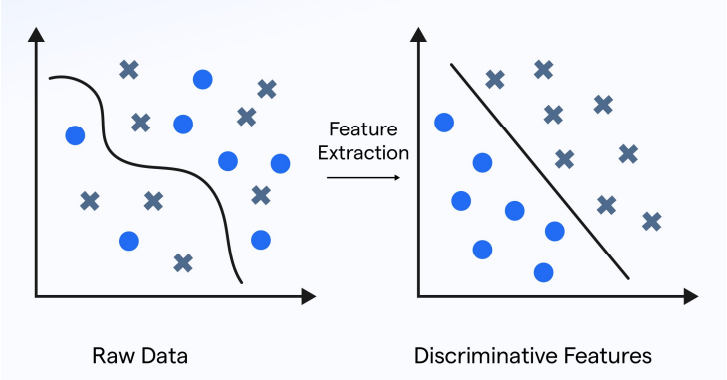
Fitur tekstur dalam *computer vision* merujuk pada pola spasial dari intensitas piksel dalam gambar *digital*. Ekstraksi dan analisis dalam fitur tekstur memungkinkan untuk mengidentifikasi, mengklasifikasikan, dan memahami struktur tekstur gambar. Berikut aspek penting fitur tekstur dalam *computer vision*:

a. Deskriptor statistik: fitur tekstur sering diekstraksi menggunakan statistik lokal atau global dari distribusi intensitas piksel dalam gambar yang termasuk metrik seperti *mean*, *variance*, korelasi, kontras, dan energi.
b. *Filter* spasial: merupakan *filter Gabor* atau *filter bank* tekstur yang digunakan untuk mengekstraksi informasi tentang pola frekuensi dan orientasi dalam gambar. *Filter* dapat menyoroti pola tekstur tertentu dan mengurangi informasi lain dalam suatu gambar yang ditentukan.
c. Matriks *co-ocurrence* : merupakan metode yang digunakan untuk mengukur hubungan spasial antara piksel dalam gambar. Matriks dapat mengekstraksi fitur seperti kontras, homogenitas, energi, dan korelasi, yang memberikan informasi tentang struktur suatu tekstur.
d. Transformasi *wavelet* : digunakan untuk membedah citra ke dalam domain frekuensi dan skala yang berbeda sehingga memungkinkan untuk mengekstraksi fitur tekstur pada berbagai tingkat resolusi dan memahami tekstur pada berbagai skala.
e. Penggunaan histogram: merupakan distribusi frekuensi dari intensitas tekstur dalam gambar yang dapat digunakan untuk mengidentifikasi pola tekstur yang berbeda dalam gambar.

f. Penggunaan pola: fitur tekstur juga dapat diekstraksi dari pola repetitif atau struktur spasial dalam gambar. Pola tekstur seperti titik, garis, atau kotak dapat digunakan sebagai fitur untuk membedakan dan mengklasifikasikan objek dalam suatu gambar tertentu.
g. Segmentasi tekstur: fitur tekstur sering dimanfaatkan pada segmentasi citra yang membedakan area dengan tekstur yang berbeda dalam gambar sehingga membantu memisahkan objek dari latar belakang atau dalam mengidentifikasi daerah dengan karakteristik tekstur tertentu.

Fitur tekstur memainkan peran penting dalam analisis citra *digital* karena membawa informasi tentang struktur dan pola dalam gambar. Penggunaan fitur tekstur membutuhkan aktivitas yang dapat mengidentifikasi dan memahami tekstur dalam gambar yang penting dalam berbagai aplikasi seperti deteksi pola, klasifikasi citra, pengenalan objek, dan banyak lagi.

Kombinasi dari ketiga jenis fitur warna, bentuk dan tekstur akan digunakan dalam aplikasi pengolahan citra yang lebih kompleks untuk menambah kemampuan analisis dan interpretasi gambar. Penerapan dalam pengenalan objek melalui fitur warna, bentuk, dan tekstur yang digunakan bersama untuk meningkatkan akurasi pengenalan objek dalam berbagai konteks *visual*.



Gambar 6. *Feature Extraction*

DAFTAR PUSTAKA

Davies, E.R. (2012) *Computer and Machine Vision Theory, Algorithms, Practicalities*. Waltham: Elsevier Inc.

Diantarakita and Rahman, A.W.W.M.A. (2019) 'Ekstraksi Ciri pada Klasifikasi Tipe Kulit Wajah Menggunakan Metode Local Binary Pattern', *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 3(8), pp. 7938–7945.

Dompeipen, T.A. and Sompie, S.R.U.. (2020) 'Penerapan Computer Vision untuk Pendeteksian dan Penghitung Jumlah Manusia', *Jurnal Teknik Informatika*, 15(4), pp. 1–12.

F., M.Y., Yuwono, B. and P. Boedi, D. (2022) *Dasar Pengolahan Citra Digital*. Yogyakarta: Lembaga Penelitian dan Pengabdian kepada Masyarakat UPN Veteran Yogyakarta.

Hendryanto Edha, Sampe Hotlan Sitorus, U.R. (2020) 'Penerapan Metode Transformasi Ruang Warna Hue Saturation Intensity untuk Mendeteksi Kematangan Buah Mangga Harum Manis', *Coding : Jurnal Komputer dan Aplikasi*, 08(1), pp. 1–10.

Hidayat, R. *et al.* (2021) 'Penerapan Metode Pembelajaran Menggunakan Ekstraksi Fitur dan Algoritma Klasifikasi untuk Identifikasi Pengenalan Iris', *Jurnal Teknik Komputer AMIK BSI*, 7(2), pp. 151–157.

Komputer, V. and Segmentasi, D.A.N. (2010) 'Implementasi Visi Komputer dan Segmentasi Citra untuk Klasifikasi Bobot Telur Ayam Ras', *Seminar Nasional Aplikasi Teknologi Informasi 2010 (SNATI 2010)*, 2010(Snati), pp. 1–5.

Manakitsa, N. *et al.* (2024) 'A Review of Machine Learning and Deep Learning for Object Detection, Semantic

Segmentation, and Human Action Recognition in Machine and Robotic Vision', *Technologies*, 12(2).

Nabusa, Y.N. (2019) 'Pengolahan Citra Digital Perbandingan Metode Histogram Equalization Dan Spesification Pada Citra Abu-Abu', *J-Icon*, 7(1), pp. 87–95.

Prabowo, D.A., Abdullah, D. and Ari Manik (2018) 'Deteksi dan Perhitungan Objek Berdasarkan Warna Menggunakan Color Object Tracking', *Pseudocode*, 5(2), pp. 85–91.

Subur, J. *et al.* (2024) 'Pemanfaatan Teknologi Computer Vision untuk Deteksi Ukuran Ikan Bandeng dalam Membantu Proses Sortir Ikan', *Cyclotron*, 7(01), pp. 52–60.

Wahyudi, T. (2023) 'Studi Kasus Pengembangan dan Penggunaan Artificial Intelligence (AI) Sebagai Penunjang Kegiatan Masyarakat Indonesia', *Indonesian Journal on Software Engineering (IJSE)*, 9(1), pp. 28–32.

Yuhandri, Y., R., A. and Syahputra, H. (2022) 'Pengenalan Teknologi Pengolahan Citra Digital', *Community Development Journal : Jurnal Pengabdian Masyarakat*, 3(2), pp. 1239–1244.

BAB 3
MACHINE LEARNING

Oleh Ayunda Kusuma Wardani, S.Tr.Kom

1. Konsep Dasar Machine Learning

Machine learning adalah sebuah cabang dalam ilmu komputer yang memungkinkan sistem komputer untuk belajar dari data (Mahesh, 2018). Dengan kata lain, mesin-mesin dapat mengenali pola-pola yang tersembunyi dalam data dan menggunakan pengetahuan tersebut untuk membuat keputusan atau melakukan tugas tertentu. Ini berarti bahwa mesin dapat mengenali tren, membuat prediksi, dan mengambil keputusan dengan tingkat akurasi yang semakin tinggi seiring bertambahnya pengalaman dan data yang diproses.

Dalam dunia perangkat lunak, machine learning memainkan peran kunci bagi pengguna dengan menyediakan solusi yang lebih personal dan efisien. Lebih jauh lagi, dalam konteks computer vision, machine learning memungkinkan sistem untuk belajar dari data visual seperti pengenalan wajah, deteksi objek, dan analisis citra medis. Dengan demikian, machine learning telah merubah paradigma interaksi manusia dengan teknologi dan membuka kesempatan bagi penelitian dan inovasi yang lebih baik lagi.

Machine learning adalah pilar utama dalam perkembangan Artificial Intelligence (AI) yang memungkinkan sistem AI untuk belajar, beradaptasi, dan memberikan solusi cerdas dalam berbagai konteks. Artificial Intelligence sendiri

dapat didefinisikan sebagai bidang ilmu yang bertujuan mengembangkan sistem komputer cerdas. Deep Learning (DL), bagian dari machine learning yang bekerja dengan menggali lapisan-lapisan pemrosesan informasi non-linier untuk mengekstrak fitur-fitur yang kompleks, memungkinkan sistem untuk belajar dari data yang lebih rumit dan meningkatkan kemampuan pemrosesan informasi secara signifikan (Woschank dkk., 2020). Dengan demikian, Deep learning merupakan level yang lebih tinggi dari machine learning dengan kemampuan belajar mandiri dari data yang lebih kompleks dan abstrak.

Sejarah pembelajaran mesin dimulai pada tahun 1943 dengan model matematika pertama jaringan saraf oleh Walter Pitts dan Warren McCulloch. Pada tahun 1949, Donald Hebb menerbitkan buku yang menjadi landasan penting dalam pengembangan machine learning dengan teori tentang hubungan perilaku dengan aktivitas otak. Alan Turing kemudian mengusulkan Tes Turing pada tahun 1950 untuk menentukan kecerdasan komputer. Pada 1950an, Arthur Samuel dari IBM mengembangkan program catur yang menggunakan teknik-teknik baru seperti pemangkasan alfa-beta dan strategi minimax, yang akhirnya membentuk dasar pembelajaran mesin. Samuel juga menciptakan istilah "machine-learning" pada tahun 1952 (Kufel dkk., 2023). Saat ini machine learning merupakan aspek penting dalam berbagai hal, termasuk penelitian. Machine learning menggunakan algoritma untuk membantu sistem komputer dalam meningkatkan kinerjanya secara progresif.

2. Pembelajaran Terawasi

Pembelajaran terawasi (Supervised Learning) adalah metode machine learning yang melibatkan pelatihan model

menggunakan kumpulan data berlabel(Sharma, 2020). Kumpulan data ini terdiri dari contoh input dan output yang diinginkan (target). Algoritma mempelajari hubungan antara input dan output dengan menganalisis data pelatihan. Setelah dilatih, model dapat digunakan untuk memprediksi output untuk data baru yang belum diketahui.

Beberapa contoh pembelajaran terawasi yang mengambil pola untuk memperkirakan nilai adalah klasifikasi, regresi, prediksi, dan boosting. Pembelajaran ini biasanya diterapkan pada aplikasi yang data sebelumnya mengantisipasi kejadian di masa depan. Ada banyak contoh pembelajaran mesin yang diawasi seperti Naïve Bayes, Pohon Keputusan, dll.

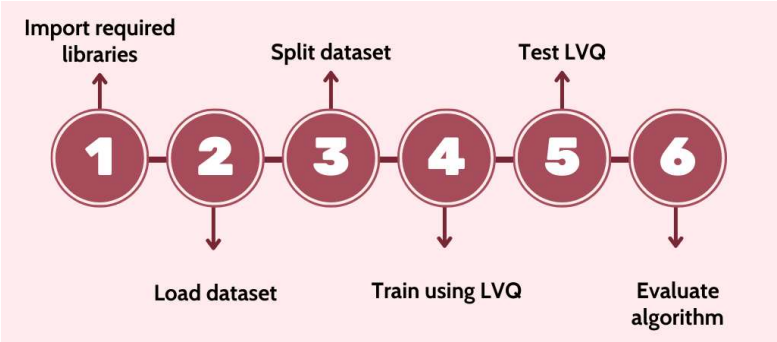
Analogi:

Seorang guru yang mengajarkan anak kecil untuk mengenali perbedaan antara apel dan jeruk. Guru menunjukkan kepada anak berbagai gambar apel dan jeruk, serta menjelaskan ciri-ciri visual masing-masing, seperti warna, bentuk, dan tekstur kulit. Anak tersebut belajar dengan melihat banyak contoh apel dan jeruk, serta mendengar penjelasan anda tentang setiap gambar. Ketika anak tersebut diberi gambar baru yang belum pernah dilihat sebelumnya, dia menggunakan pengetahuan yang telah dipelajarinya untuk mengidentifikasi apakah gambar tersebut adalah apel atau jeruk. Dalam hal ini, anak tersebut berperan seperti model pembelajaran mesin terawasi, dengan gambar-gambar apel dan jeruk beserta ciri-ciri visualnya sebagai data masukan, dan kemampuan mengidentifikasi buah sebagai variabel target.

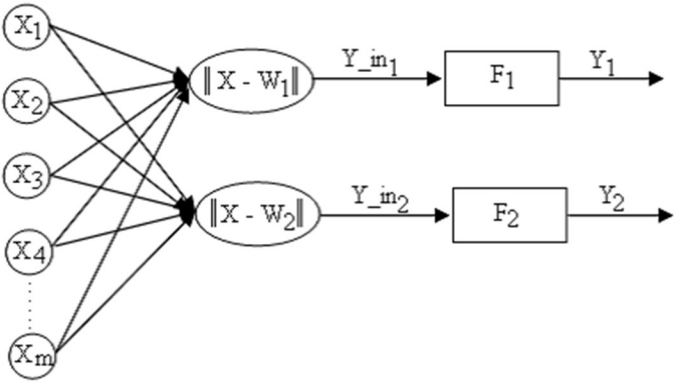
a. Neural Network

Neural network adalah salah satu teknik machine learning yang terinspirasi oleh otak manusia, mampu mempelajari hubungan antara input dan output dari data

berlabel. Neural network terdiri dari banyak neuron yang saling terhubung, dengan bobot pada setiap koneksi yang menentukan kekuatan pengaruhnya (Ridho dkk., 2022). Neural network belajar dengan menyesuaikan bobot koneksi antar neuron berdasarkan data pelatihan, sehingga mampu menangani data yang kompleks dan non-linear, serta mempelajari pola yang rumit. Neural network menjadikannya alat yang berharga untuk berbagai aplikasi, seperti pengenalan gambar, terjemahan bahasa, dan analisis sentimen. Salah satu algoritma neural network adalah Learning Vector Quantization (LVQ) merupakan sebuah metode pelatihan yang digunakan pembelajaran pada suatu lapisan kompetitif dengan pengawasan (supervised learning), dimana arsitektur jaringannya terdiri dari 1 lapisan (single layer). Kelas-kelas yang dihasilkan oleh lapisan kompetitif bergantung pada suatu jarak antara vektor-vektor input. Jika 2 vektor input memiliki kemiripan yang tinggi, lapisan kompetitif akan mengelompokkan ke-2 vektor input ke dalam kelas yang sama (Aren dkk., 2022). Secara keseluruhan, algoritma ini bekerja dengan mengelompokkan vektor input berdasarkan kemiripannya dan meminimalkan kesalahan klasifikasi.



Gambar 7. Step by Step LVQ



Gambar 8. Arsitektur LVQ

Langkah-langkah algoritma pelatihan LVQ terdiri atas (Wuryandari, 2012):

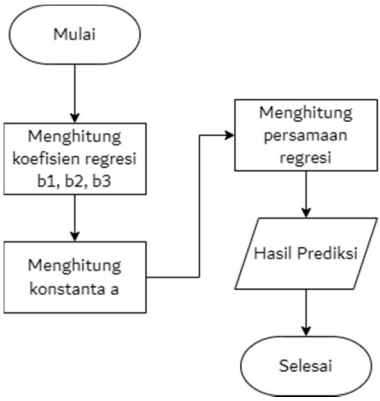
1. Inisialisasi bobot awal (W) dan parameter LVQ, yaitu maxEpoch, α, decα dan minα.
2. Masukkan sebuah data input (X) dan kelas target (T).
3. Tetapkan kondisi awal: epoch = 0.
4. Kerjakan jika: (epoch < maxEpoch) dan (α ≥ minα).
   a) epoch = epoch+1.
   b) Tentukan J sedemikian hingga ‖XiWj‖ minimal menggunakan perhitungan rumus jarak euclidian. $D(j) = \sum(Wij - x\ i)^2$
   c) Perbaiki Wj dengan ketentuan:
      Jika T = Cj maka
      Wj(t + 1) = wj (t) α (t)[x(t) – wj
      Jika T≠Cj
      maka Wj(t + 1) = wj (t) + α (t)[x(t)– wj(t)] (3)
   d) Kurangi nilai α dengan:

$$\alpha = \alpha - \alpha * \text{Dec}\alpha \quad (3)$$

5. Tes kondisi berhenti dengan keluaran berupa bobot optimal.

b. Regresi

Regresi dalam machine learning adalah serangkaian teknik dan prinsip yang digunakan untuk memodelkan dan menganalisis hubungan antara variabel terikat (output atau target) dan 1 atau lebih variabel bebas (input atau fitur) (Schober & Vetter, 2021). Tujuan utamanya adalah menemukan fungsi yang dapat memprediksi nilai variabel terikat berdasarkan variabel bebas. Model regresi, seperti regresi linier, polinomial, dan ridge, digunakan untuk tugas prediksi dengan output nilai kontinu dan berusaha meminimalkan kesalahan prediksi melalui teknik optimasi dan penyesuaian parameter. Model ini dibangun berdasarkan asumsi kausalitas, dengan penentuan variabel bebas dan terikat berdasarkan teori, penelitian sebelumnya, atau penjelasan logis.



Gambar 9 Flowchart Regresi

Rumus regresi:

$$Y = a+bX$$

Ket :

Y = variable kriterium
X = variabel predictor
a = variabel konstan
b= koefisien arah regresi linier

Dimana harga a dan b sebagai berikut :

a = $\frac{(\sum Y)(\sum X^2)-(\sum X)(\sum XY)}{n\sum X^2-(\sum X)^2}$ b = $\frac{n\sum XY-(\sum X)(\sum Y)}{n\sum X^2-(\sum x)^2}$

3. Pembelajaran Tidak Terawasi

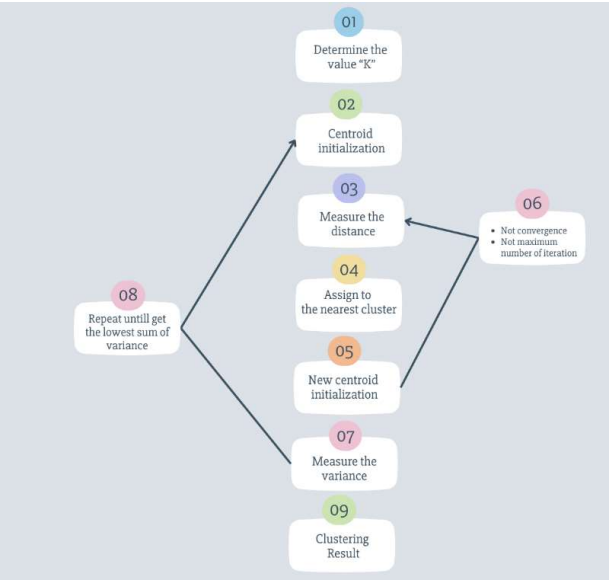
Pembelajaran tidak terawasi (Unsupervised learning) adalah jenis pembelajaran mesin di mana algoritma digunakan untuk mengidentifikasi pola atau struktur pada data yang tidak mempunyai label atau kategori yang telah ditentukan sebelumnya (Sathya Professor dkk., 2013). Sehingga pembelajaran tidak terawasi ini mengatur dirinya sendiri belajar menggunakan algoritma untuk menemukan pola yang tersembunyi pada data masukan yang tanpa label, tanpa adanya sinyal kesalahan untuk mengevaluasi solusi. Kurangnya arahan dalam pembelajaran terawasi memungkinkan algoritma mengidentifikasi pola-pola baru yang tidak terduga.

Analogi:
Bayangkan unsupervised learning seperti menngindentifikasi koleksi foto tanpa deskripsi. Anda memiliki sekumpulan gambar yang belum diberi label, dan tugasnya adalah menemukan pola di dalamnya tanpa petunjuk eksternal. Algoritma unsupervised learning akan mencoba mengelompokkan gambar berdasarkan kemiripan visual,

membantu mengidentifikasi pola dan struktur dalam dataset tanpa bantuan label. Ini mirip dengan menemukan tema atau kesamaan visual di antara gambar-gambar tanpa penjelasan tambahan.

a. K-Means

Algoritma K-Means merupakan metode pengelompokan data yang mempartisi data menjadi beberapa cluster, dimana data yang serupa dikelompokkan dalam satu cluster yang sama dan data yang berbeda dikelompokkan dalam cluster yang lain (Prasetyo dkk., t.t.). K-means biasanya digunakan untuk berbagai jenis masalah clustering dalam banyak bidang seperti segmentasi pelanggan, kompresi gambar, deteksi anomali, rekomendasi produk, dan lain-lain.



Gambar 10. Step by Step K-means

Rumus Umum K-Means:

1. Menentukan centroid tiap kelompok dihitung sebagai rata-rata nilai setiap fitur. Dengan M jumlah data dan i fitur ke-i:

$$C_i = \frac{1}{M}\sum_{j=1}^{M} X_j$$

2. Hitung jarak titik terdekat:

$$D(x_2, x_1,) = \sqrt{\sum_{j=1}^{P} | X_{2j} - X_{1j} |^2}$$

3. Pengalokasian keanggotaan titik:

$$a_{ji} = \begin{cases} 1, & d = \min(D(X_j, C_i)) \\ 0, & lainnya \end{cases}$$

4. Fungsi Objektif:

$$F = \Sigma_{j=1}^{N} \Sigma_{i=1}^{K} a_{ji}(X_j, C_i)$$

Beberapa contoh penerapan K-Means dalam kegiatan sehari-hari:

a) Analisis Jaringan Sosial:

K-Means dapat digunakan untuk mengidentifikasi komunitas dan kelompok influencer dalam jaringan media sosial. Informasi ini dapat digunakan oleh perusahaan untuk menargetkan iklan, melacak tren, dan mengelola reputasi online mereka.

b) Analisis Data Sensor:

K-Means dapat digunakan untuk mendeteksi anomali dan pola dalam data sensor yang dikumpulkan dari peralatan industri dan infrastruktur. Ini dapat membantu

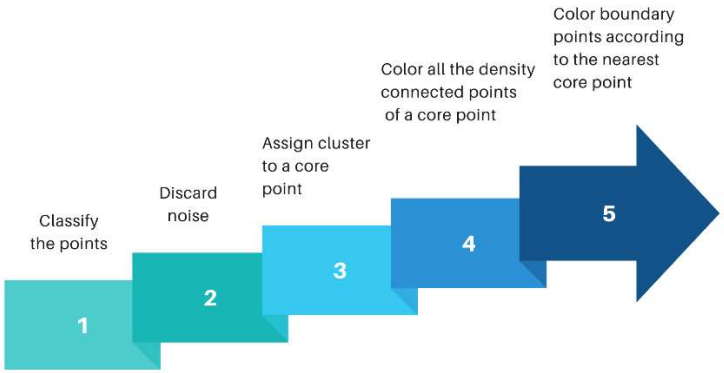
memprediksi kegagalan peralatan, dan meningkatkan efisiensi operasional.

c) Analisis Data Cuaca:

K-Means dapat digunakan untuk mengidentifikasi pola cuaca berdasarkan suhu, tekanan udara, curah hujan, atau faktor lainnya. Informasi ini dapat membantu ahli meteorologi memprediksi cuaca, mengeluarkan peringatan dini, dan mempersiapkan bencana alam.

b. DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) adalah algoritma yang proses pembentukan clusternya dilakukan berdasarkan tingkat kepadatan jarak antar objek pada dataset. DBSCAN mempunyai kelebihan yaitu mampu mendeteksi noise (Bariklana & Fauzan, 2023). Tidak seperti K-means, yang membutuhkan jumlah cluster yang harus ditentukan sebelumnya, DBSCAN dapat menemukan jumlah cluster yang optimal secara otomatis berdasarkan kepadatan data.



Gambar 11. Step by Step DBSCAN

Beberapa contoh penerapan DBSCAN dalam kegiatan sehari-hari:

1) Deteksi Anomali pada Jaringan Keamanan:

DBSCAN ini bisa digunakan untuk pendeteksi aktivitas mencurigakan pada jaringan komputer. Aktivitas yang normal akan membentuk cluster yang padat, sedangkan aktivitas anomali (seperti percobaan hacking) akan berada di luar cluster tersebut sebagai noise.

2) Analisis Sosial Media:

Dalam analisis data sosial media, DBSCAN dapat digunakan untuk mengidentifikasi kelompok pengguna yang berinteraksi secara intensif (misalnya, sekelompok orang yang sering berkomentar atau like pada posting yang sama), serta mendeteksi outlier seperti akun spam.

3) Sistem Navigasi dan Transportasi:

Dalam sistem navigasi dan transportasi, DBSCAN digunakan untuk mengidentifikasi pola lalu lintas, seperti deteksi cluster kendaraan yang berhenti dalam lalu lintas macet. Ini dapat membantu dalam pengaturan lalu lintas dan pengambilan keputusan untuk mengurangi kemacetan.

4) Rekomendasi Produk:

Dalam e-commerce, DBSCAN dapat digunakan untuk mengidentifikasi kelompok pelanggan dengan pola pembelian yang serupa. Informasi ini dapat digunakan untuk memberikan rekomendasi produk yang lebih relevan bagi setiap kelompok pelanggan.

5) Deteksi Penipuan Kartu Kredit:

DBSCAN dapat digunakan untuk mengidentifikasi transaksi kartu kredit yang tidak biasa, membantu mendeteksi penipuan dan aktivitas mencurigakan. Hal ini

dapat membantu melindungi konsumen dari pencurian identitas dan kerugian finansial.

6) Pengelompokan Spesies Hewan:

DBSCAN dapat digunakan untuk mengelompokkan spesies hewan berdasarkan karakteristik fisik dan habitat mereka, membantu memahami keanekaragaman hayati dan hubungan ekologis. Informasi ini dapat digunakan untuk upaya konservasi dan pengelolaan sumber daya alam.

DAFTAR PUSTAKA

Aren, G., Dengan, A., Melisa, C., Manaor, A., Pardede, H., & Sihombing, M. (2022). Implementasi Learning Vector Quantization (LVQ) Dalam Mengidentifikasi. Dalam Sci-Tech Journal (Vol. 1, Nomor 1).

Bariklana, M., & Fauzan, A. (2023). IMPLEMENTATION OF THE DBSCAN METHOD FOR CLUSTER MAPPING OF EARTHQUAKE SPREAD LOCATION. BAREKENG: Jurnal Ilmu Matematika dan Terapan, 17(2), 0867–0878. https://doi.org/10.30598/barekengvol17iss2pp0867-0878

Kufel, J., Bargieł-Łączek, K., Kocot, S., Koźlik, M., Bartnikowska, W., Janik, M., Czogalik, Ł., Dudek, P., Magiera, M., Lis, A., Paszkiewicz, I., Nawrat, Z., Cebula, M., & Gruszczyńska, K. (2023). What Is Machine Learning, Artificial Neural Networks and Deep Learning?—Examples of Practical Applications in Medicine. Dalam Diagnostics (Vol. 13, Nomor 15). Multidisciplinary Digital Publishing Institute (MDPI). https://doi.org/10.3390/diagnostics13152582

Mahesh, B. (2018). Machine Learning Algorithms-A Review. International Journal of Science and Research. https://doi.org/10.21275/ART20203995

Prasetyo, M., Ansori, N. A., Firdausi, Y., & Wahid, M. F. (t.t.). Implementation of K-Means Clustering for Analysis Students English Proficiency. Dalam Science and Technology | (Vol. 1, Nomor 1).

Ridho, I. I., Mahalisa, G., Sari, D. R., Fikri, I., Kalimantan, I., Al, M. A., Banjarmasin, B., Informasi, F. T., Islam, U., Muhammad, K., & Al, A. (2022). METODE NEURAL NETWORK UNTUK PENENTUAN AKURASI PREDIKSI

HARGA RUMAH. Dalam Technologia (Vol. 13, Nomor 1).

Sathya Professor, R., Nivas College, J., & Abraham Professor, A. (2013). Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification. Dalam IJARAI) International Journal of Advanced Research in Artificial Intelligence (Vol. 2, Nomor 2). www.ijarai.thesai.org

Schober, P., & Vetter, T. R. (2021). Statistical Minute Logistic Regression in Medical Research (Vol. 132, Nomor 2). www.anesthesia-analgesia.org365

Sharma, R. (2020). Study of Supervised Learning and Unsupervised Learning. International Journal for Research in Applied Science and Engineering Technology, 8(6), 588–593. https://doi.org/10.22214/ijraset.2020.6095

Woschank, M., Rauch, E., & Zsifkovits, H. (2020). A review of further directions for artificial intelligence, machine learning, and deep learning in smart logistics. Sustainability (Switzerland), 12(9). https://doi.org/10.3390/su12093760

BAB 4
PEMROGAMAN PYTHON

Oleh Victor Benny Alexsius Pardosi, S.Kom., M.Sc.

1. Dasar Bahasa Python

a. Pengenalan Python

Python adalah bahasa pemrograman yang dikembangkan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991. Bahasa ini dinamakan "Python" bukan karena ular, tetapi karena Guido adalah penggemar acara komedi Inggris "Monty Python's Flying Circus." Sejak dirilis, Python telah menjadi salah satu bahasa pemrograman paling populer di dunia, digunakan dalam berbagai bidang seperti pengembangan web, ilmu data, kecerdasan buatan, otomasi skrip, dan banyak lagi.

Berikut beberapa keunggulan Python:

- Mudah Dipelajari dan Digunakan: Sintaks Python yang bersih dan mudah dipahami menjadikannya bahasa yang ideal untuk pemula. Anda dapat menulis kode yang jelas dan mudah dibaca dengan Python, yang meningkatkan produktivitas pengembang.
- Bahasa Serbaguna: Python mendukung berbagai paradigma pemrograman termasuk pemrograman berorientasi objek, prosedural, dan fungsional. Hal ini membuatnya fleksibel dan dapat digunakan untuk berbagai jenis proyek.
- Komunitas yang Besar: Python memiliki komunitas pengguna dan pengembang yang sangat aktif. Ini

berarti banyak sumber daya, dokumentasi, dan dukungan yang tersedia secara gratis.
- Library dan Framework yang kuat: Python memiliki ekosistem library dan framework yang sangat kaya, seperti Django dan Flask untuk pengembangan web, NumPy dan Pandas untuk ilmu data, TensorFlow dan Keras untuk pembelajaran mesin, serta OpenCV untuk visi komputer.
- Portabilitas: Kode Python dapat dijalankan di berbagai sistem operasi tanpa perubahan berarti, seperti Windows, macOS, dan Linux.
- Interaktif dan Interpreted: Python adalah bahasa yang diinterpretasi, yang berarti Anda bisa langsung menjalankan kode tanpa perlu proses kompilasi. Ini memudahkan debugging dan pengembangan cepat.

Berikut beberapa contoh penggunaan Python dalam Berbagai Bidang:

- Pengembangan Web: Framework seperti Django dan Flask memungkinkan pengembangan aplikasi web yang cepat dan aman.
- Data Science dan Analisis: Library seperti Pandas, NumPy, dan Matplotlib menjadikan Python sebagai pilihan utama untuk analisis data dan visualisasi.
- Kecerdasan Buatan dan Pembelajaran Mesin: Dengan Library seperti TensorFlow, Keras, dan scikit-learn, Python adalah bahasa utama yang digunakan untuk pengembangan aplikasi AI dan machine learning.
- Otomasi dan Scripting: Python sering digunakan untuk menulis skrip yang mengotomatisasi tugas-tugas yang berulang.

- Pengembangan Permainan: Library seperti Pygame memungkinkan pengembangan permainan 2D yang interaktif.

Dengan semua keunggulan dan kegunaannya, tidak mengherankan bahwa Python menjadi bahasa pilihan bagi banyak pengembang di seluruh dunia. Dalam bab ini, kita akan menjelajahi dasar-dasar Python, yang akan memberikan fondasi kuat untuk mempelajari aplikasi lebih lanjut dalam machine learning dan computer vision.

b. Instalasi Python

Untuk mulai menggunakan Python, langkah pertama adalah menginstalnya di sistem operasi Anda. Berikut adalah panduan langkah demi langkah untuk menginstal Python di Windows, macOS, dan Linux.

➢ Instalasi Python di Windows

1. Unduh Installer Python
   - Kunjungi situs resmi Python di python.org.
   - Buka halaman Downloads dan unduh installer Python versi terbaru yang sesuai dengan sistem operasi Windows Anda (32-bit atau 64-bit).
2. Jalankan Installer
   - Setelah unduhan selesai, buka file installer.
   - Centang opsi "Add Python to PATH" di bagian bawah jendela installer. Ini penting agar Anda dapat menjalankan Python dari command line.
   - Klik "Install Now" untuk memulai proses instalasi.
3. Verifikasi Instalasi
   - Buka Command Prompt (Anda dapat mencarinya di menu Start).

- Ketik *python --version* dan tekan Enter. Jika Python terinstal dengan benar, Anda akan melihat versi Python yang terpasang.

➢ Instalasi Python di macOS

1. Gunakan Homebrew (opsional):
   - Homebrew adalah manajer paket untuk macOS yang memudahkan instalasi berbagai software.
2. Instal Python
   - Unduh installer dari situs resmi Python di python.org dan jalankan file installer.
   - Atau, jika menggunakan Homebrew, buka Terminal dan jalankan perintah berikut: *brew install python*
3. Verifikasi Instalasi
   - Buka terminal, ketik *python3 --version* dan tekan Enter. Jika Python terinstal dengan benar, Anda akan melihat versi Python yang terpasang.

➢ Instalasi Python di Linux

1. Menggunakan Package Manager
   - Hampir semua distribusi Linux sudah dilengkapi dengan Python secara default. Namun, jika Anda perlu menginstal atau memperbarui Python, Anda dapat menggunakan package manager yang sesuai dengan distribusi, Silahkan gunakan command dibawah ini:
   - Debian/Ubuntu: *apt install python3*
   - Fedora: *dnf install python3*
   - Arch Linux: *pacman -S python*

2. Verifikasi Instalasi
   - Buka terminal, ketik *python3 --version* dan tekan Enter. Jika Python terinstal dengan benar, Anda akan melihat versi Python yang terpasang

Setelah instalasi berhasil, Anda dapat menjalankan interpreter Python dengan mengetik *python* atau *python3* (tergantung pada sistem operasi dan versi Python yang diinstal) di Command Prompt atau Terminal. Anda akan masuk ke mode interaktif Python, yang ditandai dengan prompt >>>. Untuk keluar dari mode interaktif, Anda bisa mengetik *exit()* dan tekan Enter.

2. Sintaks Dasar

Sintaks Python sangat sederhana dan mudah dipelajari. Dalam bagian ini, kita akan membahas elemen-elemen dasar dari bahasa Python yang akan sering digunakan dalam penulisan kode.

a. Variabel dan Tipe Data

Python adalah bahasa yang dinamis, artinya Anda tidak perlu mendeklarasikan tipe variabel secara eksplisit. Anda cukup memberikan nilai kepada variabel, dan Python akan menangani sisanya. Contoh penulisan variabel dalam Python: nama = "Victor" (string), usia = 33 (integer), tinggi = 1.70 (float), suka_menulis = True (boolean).

b. Operator

Python mendukung berbagai jenis operator, termasuk operator aritmatika, perbandingan, logika, dan penugasan.

- Operator aritmatika meliputi: penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/),

modulus (sisa bagi, %), eksponen (pangkat, **), dan pembagian bulat (//).
    - Operator perbandingan meliputi: sama dengan (==), tidak sama dengan (!=), lebih besar (>), lebih kecil (<), lebih besar atau sama dengan (>=), lebih kecil atau sama dengan (<=).
- Operator logika meliputi: AND (and), OR (or), NOT (not).
- Operator penugasan meliputi: penugasan tambah (+=), penugasan kurang (-=), penugasan kali (*=), penugasan bagi (/=), penugasan modulus (%=), penugasan eksponen (**=), penugasan pembagian bulat (//=).

c. Kontrol Alur (If-Else, Loop)
Untuk kontrol alur, Python menggunakan pernyataan if-else dan loop.
- Pernyataan if-else digunakan untuk menjalankan kode berdasarkan kondisi tertentu. Contoh penulisan if-else: nilai = 85; if nilai >= 90: print("Grade A"); elif nilai >= 80: print("Grade B"); elif nilai >= 70: print("Grade C"); else: print("Grade D").
- Loop digunakan untuk menjalankan kode berulang kali. Ada dua jenis loop dalam Python, yaitu for loop dan while loop. Contoh for loop: for i in range(5): print(i). Contoh while loop: count = 0; while count < 5: print(count); count += 1.

3. Fungsi dan Modul

Python memungkinkan Anda untuk mendefinisikan fungsi dan mengimpor modul.

- Fungsi didefinisikan menggunakan kata kunci def. Contoh mendefinisikan dan memanggil fungsi: def greet(nama): print(f"Hello, {nama}!"); greet("Pardosi").
- Modul dapat diimpor dan digunakan dalam kode Anda. Contoh mengimpor modul math dan menggunakan fungsi sqrt: import math; print(math.sqrt(16)). Anda juga dapat mengimpor bagian tertentu dari modul. Contoh mengimpor konstanta pi: from math import pi; print(pi).

Dengan memahami dasar-dasar sintaks Python ini, Anda sudah siap untuk mulai menulis dan menjalankan kode Python yang lebih kompleks. Pada bagian berikutnya, kita akan membahas tentang IDE dan text editor yang dapat Anda gunakan untuk mengembangkan proyek Python Anda.

4. IDE/Text Editor Python (Online dan Offline)

Untuk menulis dan mengembangkan kode Python, Anda memerlukan lingkungan pengembangan yang nyaman dan efisien. Ada berbagai IDE (Integrated Development Environment) dan text editor yang dapat digunakan, baik secara online maupun offline.

IDE adalah perangkat lunak yang menyediakan fasilitas komprehensif untuk pengembangan perangkat lunak. IDE biasanya mencakup editor kode, debugger, dan build automation tools. Text editor adalah tool yang lebih ringan yang berfokus pada pengeditan teks, tetapi dapat ditambahkan plugin untuk mendukung pengembangan perangkat lunak.

a. IDE Populer untuk Python

1. PyCharm
   PyCharm adalah salah satu IDE yang paling populer untuk pengembangan Python, dikembangkan oleh JetBrains. PyCharm menyediakan fitur-fitur seperti code completion, debugging, refactoring, dan integrasi dengan sistem kontrol versi. Untuk menginstal PyCharm, Anda bisa mengunduhnya dari situs resmi JetBrains dan mengikuti petunjuk instalasi yang diberikan. Setelah terinstal, Anda bisa memulai proyek baru dengan memilih opsi "New Project" dan mengatur interpreter Python yang akan digunakan.
2. Visual Studio Code (VS Code)
   Visual Studio Code adalah editor kode yang dikembangkan oleh Microsoft, yang mendukung berbagai bahasa pemrograman termasuk Python. VS Code dapat diperluas dengan berbagai ekstensi untuk menambahkan fungsionalitas seperti linting, debugging, dan integrasi dengan Jupyter Notebook. Untuk menginstal VS Code, unduh dari situs resmi Visual Studio Code dan ikuti petunjuk instalasi. Setelah itu, Anda dapat menginstal ekstensi Python dari marketplace VS Code untuk mulai mengembangkan proyek Python.

b. Text Editor Populer

1. Sublime Text
   Sublime Text adalah text editor yang ringan dan cepat, dengan dukungan untuk berbagai bahasa pemrograman termasuk Python. Anda dapat menambahkan plugin melalui Package Control untuk menambahkan fitur tambahan. Untuk menginstal Sublime Text, unduh dari situs resmi Sublime Text dan

ikuti petunjuk instalasi. Setelah itu, Anda bisa menambahkan plugin seperti "Anaconda" untuk meningkatkan dukungan Python.

2. Atom
   Atom adalah text editor open-source yang dikembangkan oleh GitHub, yang juga mendukung berbagai bahasa pemrograman. Fitur Atom dapat ditambah dengan menginstall berbagai paket untuk menambahkan fungsionalitas tambahan. Untuk menginstal Atom, unduh dari situs resmi Atom dan ikuti petunjuk instalasi. Setelah itu, Anda bisa menambahkan paket seperti "ide-python" untuk meningkatkan dukungan Python.

c. IDE Online untuk Python

1. Google Colab
   Google Colab adalah layanan berbasis cloud yang memungkinkan Anda menulis dan menjalankan kode Python di browser. Colab sangat cocok untuk proyek-proyek yang memerlukan komputasi berat seperti machine learning, karena Anda dapat menggunakan GPU dan TPU gratis dari Google. Anda bisa mengakses Google Colab melalui situs colab.research.google.com dan mulai menulis kode Python dalam notebook.
2. Repl.it
   Repl.it adalah platform online yang mendukung berbagai bahasa pemrograman, termasuk Python. Repl.it menyediakan editor kode, terminal, dan dukungan untuk kolaborasi secara real-time. Anda bisa mengakses Repl.it melalui situs repl.it dan mulai membuat repl (proyek) baru untuk menulis dan menjalankan kode Python.
3. Jupyter Notebook

Jupyter Notebook adalah aplikasi web yang memungkinkan Anda membuat dan berbagi dokumen yang berisi kode live, persamaan, visualisasi, dan teks naratif. Jupyter Notebook sangat populer di kalangan data scientist dan researcher. Anda bisa menginstal Jupyter Notebook secara lokal atau menggunakan layanan online seperti Google Colab. Untuk menginstal Jupyter Notebook secara lokal, Anda bisa menggunakan pip dengan perintah pip install notebook, dan kemudian menjalankannya dengan perintah jupyter notebook.

Dengan berbagai pilihan IDE dan text editor ini, Anda dapat memilih lingkungan pengembangan yang paling sesuai dengan kebutuhan dan preferensi Anda untuk mengembangkan proyek Python. Pada bagian berikutnya, kita akan membahas tentang library Python yang dapat digunakan untuk machine learning dan computer vision namun sebelum itu akan membahas praktik terbaik untuk pemrograman python.

5. Praktik Terbaik dalam Pemrograman Python

Untuk menjadi pengembang Python yang efektif, penting untuk memahami dan menerapkan praktik terbaik dalam penulisan kode. Ini termasuk penulisan kode yang rapi dan terstruktur, serta debugging dan testing.

a. Penulisan Kode yang Rapi dan Terstruktur

1. PEP 8

   PEP 8 adalah panduan gaya untuk penulisan kode Python yang menjelaskan cara menulis kode yang konsisten dan mudah dibaca. Beberapa poin utama

dari PEP 8 meliputi menggunakan indentasi 4 spasi, menjaga garis kode tidak lebih dari 79 karakter, dan menggunakan garis kosong untuk memisahkan fungsi dan kelas. Contohnya, memberi nama variabel dengan gaya snake_case seperti nama_pengguna dan kelas dengan gaya CamelCase seperti NamaKelas.

2. Docstring dan Komentar
   Menambahkan docstring dan komentar pada kode Anda membantu mendokumentasikan fungsionalitas dan tujuan dari bagian kode tertentu. Contoh docstring untuk fungsi: def hitung_luas(sisi): """Menghitung luas persegi.""" return sisi * sisi. Gunakan komentar untuk menjelaskan bagian kode yang kompleks atau langkah-langkah yang mungkin tidak jelas, misalnya # Menghitung luas persegi.

b. Debugging dan Testing

1. Debugging
   Debugging adalah proses menemukan dan memperbaiki bug dalam kode. Python menyediakan beberapa alat untuk membantu dalam debugging. Anda bisa menambahkan pernyataan print di berbagai titik dalam kode untuk memeriksa nilai variabel, misalnya print(nama_variabel). Menggunakan debugger seperti pdb dengan menambahkan import pdb; pdb.set_trace() juga sangat berguna untuk menjalankan kode baris demi baris dan memeriksa nilai variabel.

2. Testing
   Testing adalah proses memverifikasi bahwa kode Anda bekerja seperti yang diharapkan. Python menyediakan beberapa framework untuk melakukan testing, seperti

unittest dan pytest. Contoh penggunaan unittest: import unittest; class TestHitungLuas(unittest.TestCase): def test_luas(self): self.assertEqual(hitung_luas(5), 25); if __name__ == '__main__': unittest.main(). Contoh penggunaan pytest: def test_hitung_luas(): assert hitung_luas(5) == 25.

Dengan menerapkan praktik terbaik ini, Anda dapat menulis kode Python yang lebih rapi, terstruktur, dan bebas dari bug, yang pada akhirnya akan meningkatkan produktivitas dan kualitas proyek Anda. Pada bagian berikutnya, kita akan membahas tentang cara menangani error dan exception di Python.

6. Memahami Error dan Exception

Dalam pengembangan perangkat lunak, menghadapi error dan exception adalah hal yang biasa. Python menyediakan mekanisme untuk menangani error dan exception, yang sangat penting untuk membuat aplikasi yang tangguh dan bebas bug. Berikut beberapa cara menangani Error dan Exception:

1. Try-Except Block
   Untuk menangani error, Python menggunakan blok try-except. Anda menempatkan kode yang mungkin menghasilkan error di dalam blok try, dan menangani error di dalam blok except. Misalnya, try: hasil = 10 / 0 except ZeroDivisionError: print("Tidak bisa membagi dengan nol").
2. Multiple Except Blocks
   Anda dapat menangani berbagai jenis error dengan menggunakan beberapa blok except. Contohnya, try:

hasil = int("abc") except ValueError: print("Tidak bisa mengubah string ke integer") except TypeError: print("Terjadi kesalahan tipe data").

3. Else and Finally
   Blok else dapat digunakan untuk menjalankan kode jika tidak ada exception yang terjadi, sementara blok finally digunakan untuk kode yang harus dijalankan baik ada exception maupun tidak. Contohnya, try: hasil = 10 / 2 except ZeroDivisionError: print("Tidak bisa membagi dengan nol") else: print("Pembagian berhasil") finally: print("Blok finally dijalankan").
4. Raising Exceptions
   Anda juga dapat mentrigger exception secara manual dengan menggunakan pernyataan raise. Misalnya, raise ValueError("Terjadi kesalahan nilai").
5. Custom Exception
   Anda dapat mendefinisikan exception sendiri dengan membuat kelas yang diturunkan dari kelas Exception. Misalnya, class CustomError(Exception): pass; try: raise CustomError("Ini adalah custom error") except CustomError as e: print(e).

Dengan memahami cara menangani error dan exception, Anda dapat membuat kode Python yang lebih andal dan mudah di-debug. Pada bagian berikutnya, kita akan membahas tentang library Python yang dapat digunakan untuk machine learning dan computer vision.

7. Python Library

Python memiliki ekosistem library yang sangat kaya, yang memungkinkan pengembang untuk dengan mudah mengakses dan menggunakan berbagai alat dan fungsi untuk menyelesaikan berbagai tugas. Dalam konteks machine

learning dan computer vision, beberapa library yang paling sering digunakan adalah NumPy, Pandas, Matplotlib, Seaborn, Scikit-Learn, TensorFlow, Keras, dan OpenCV.

a. Manajemen Paket dengan pip

Pip adalah manajer paket untuk Python yang memungkinkan Anda untuk menginstal, mengupgrade, dan menghapus library Python. Untuk menginstal sebuah library, Anda bisa menggunakan perintah pip install nama_library. Misalnya, untuk menginstal NumPy, Anda bisa mengetik pip install numpy.

b. Library untuk Machine Learning dan Computer Vision

1. NumPy
   NumPy adalah library dasar untuk komputasi numerik di Python. Library ini menyediakan dukungan untuk array multidimensi dan berbagai fungsi matematika. Contoh penggunaan NumPy adalah membuat array dengan import numpy as np; a = np.array([1, 2, 3]).
2. Pandas
   Pandas adalah library yang digunakan untuk manipulasi dan analisis data. Library ini menyediakan struktur data seperti DataFrame, yang memungkinkan Anda untuk bekerja dengan data tabular. Contoh penggunaan Pandas adalah membaca file CSV dengan import pandas as pd; df = pd.read_csv('data.csv').
3. Matplotlib
   Matplotlib adalah library untuk visualisasi data. Library ini memungkinkan Anda untuk membuat berbagai jenis grafik seperti garis, batang, dan histogram. Contoh penggunaan Matplotlib adalah membuat grafik garis

dengan import matplotlib.pyplot as plt; plt.plot([1, 2, 3], [4, 5, 6]); plt.show().

4. Seaborn
   Seaborn adalah library visualisasi data yang dibangun di atas Matplotlib. Library ini menyediakan antarmuka tingkat tinggi untuk menggambar grafik statistik. Contoh penggunaan Seaborn adalah membuat grafik scatter plot dengan import seaborn as sns; sns.scatterplot(x=[1, 2, 3], y=[4, 5, 6]).
5. Scikit-Learn
   Scikit-Learn adalah library untuk machine learning di Python. Library ini menyediakan berbagai algoritma machine learning seperti regresi, klasifikasi, dan clustering. Contoh penggunaan Scikit-Learn adalah membuat model regresi linier dengan from sklearn.linear_model import LinearRegression; model = LinearRegression(); model.fit(X, y).
6. TensorFlow
   TensorFlow adalah library open-source untuk komputasi numerik dan machine learning yang dikembangkan oleh Google. Library ini sangat populer untuk pengembangan model deep learning. Contoh penggunaan TensorFlow adalah membuat dan melatih model neural network dengan import tensorflow as tf; model = tf.keras.Sequential([...]); model.compile(...); model.fit(X, y).
7. Keras
   Keras adalah antarmuka high-level untuk TensorFlow yang membuat pembuatan dan pelatihan model neural network menjadi lebih mudah. Contoh penggunaan Keras adalah membuat dan melatih model dengan

from tensorflow.keras.models import Sequential; model = Sequential([...]); model.compile(...); model.fit(X, y).

8. OpenCV
   OpenCV adalah library open-source untuk computer vision dan image processing. Library ini menyediakan berbagai fungsi untuk memproses gambar dan video. Contoh penggunaan OpenCV adalah membaca dan menampilkan gambar dengan import cv2; img = cv2.imread('image.jpg'); cv2.imshow('image', img); cv2.waitKey(0); cv2.destroyAllWindows().

Dengan menguasai library-library ini, Anda dapat membangun aplikasi machine learning dan computer vision yang kuat dan efisien.

Dalam Bab 4 ini, kita telah membahas dasar-dasar pemrograman Python yang mencakup pengenalan Python, instalasi Python, sintaks dasar, praktik terbaik dalam penulisan kode, debugging dan testing, serta pengelolaan dan penggunaan berbagai library Python yang penting dalam machine learning dan computer vision. Dengan pemahaman yang kuat tentang konsep-konsep ini, Anda sekarang memiliki fondasi yang kokoh untuk menjelajahi lebih lanjut dan mengembangkan aplikasi yang kompleks dan inovatif menggunakan Python. Memahami dan menerapkan teknik-teknik ini akan membantu Anda menjadi pengembang yang lebih efisien dan efektif, membuka peluang tak terbatas dalam dunia pemrograman dan teknologi.

DAFTAR PUSTAKA

Zulunov, R. and Soliev, B., 2023. Importance of Python language in development of artificial intelligence. *Потомки Аль-Фаргани*, *1*(1), pp.7-12.

Phatthiyaphaibun, W., Chaovavanich, K., Polpanumas, C., Suriyawongkul, A., Lowphansirikul, L., Chormai, P., Limkonchotiwat, P., Suntorntip, T. and Udomcharoenchaikit, C., 2023. Pythainlp: Thai natural language processing in python. *arXiv preprint arXiv:2312.04649*.

Hill, C., 2020. *Learning scientific programming with Python*. Cambridge University Press.

Saabith, S., Vinothraj, T. and Fareez, M., 2021. A review on Python libraries and Ides for Data Science. *Int. J. Res. Eng. Sci*, *9*(11), pp.36-53.

Sharma, S., Rattan, P., Lakshmipathi, B., Kumar, D., Kumar, S. and Ramakrishna, S., 2024. Automatic Error Detection Using Python. In *Computer Science Engineering and Emerging Technologies* (pp. 501-507). CRC Press.

Tonggiroh, M., Pardosi, V.B.A., Basiroh, B. and Nugroho, F., REKAYASA PERANGKAT LUNAK.

Kong, Q., Siauw, T. and Bayen, A., 2020. *Python programming and numerical methods: A guide for engineers and scientists*. Academic Press.

Martelli, A., Ravenscroft, A.M., Holden, S. and McGuire, P., 2023. *Python in a Nutshell*. " O'Reilly Media, Inc.".

BAB 5
PERSIAPAN DAN PRA PEMROSESAN DATA *(Data Preparation and Preprocessing)*

Oleh Qonitatul Hasanah, S.S.T., M.Tr.T.

Apakah Anda pernah menghadapi masalah dengan gambar berkualitas rendah dalam proyek machine learning atau computer vision? Sebelum melatih model atau menjalankan algoritma, seringkali diperlukan pra-pemrosesan gambar untuk hasil terbaik. Ini mempercepat waktu pelatihan model dan inferensi, misalnya dengan mengurangi ukuran gambar.

Tujuan utama pra-pemrosesan adalah meningkatkan data gambar dengan mengurangi distorsi yang tidak diinginkan atau meningkatkan fitur penting untuk pemrosesan lebih lanjut (Minh et al., 2018).

1. Pengumpulan Data dan Anotasi

a. Pengumpulan Data

Proses pengumpulan data gambar adalah langkah penting dalam pembangunan model pembelajaran mesin atau visi komputer yang efektif. Sumber data dapat berasal dari berbagai sumber, seperti basis data publik seperti ImageNet, COCO dataset, Kaggle, atau Open Images, pengambilan gambar langsung, perangkat perekam lain, perangkat sensor, simulasi, atau koleksi pribadi (Stojnev & Ilić, 2020).

Konsistensi dalam data, termasuk pencahayaan, latar belakang, dan sudut pengambilan gambar, krusial untuk pelatihan model. Hal ini dapat mempengaruhi:

- **Kualitas Model:** Data konsisten memungkinkan model menangkap pola yang stabil, menghasilkan keputusan yang andal.
- **Mencegah Bias:** Ketidak-konsistenan dapat mengakibatkan pembelajaran pola yang tidak relevan, mengurangi kemampuan model untuk menggeneralisasi.
- **Replikabilitas:** Data yang konsisten memastikan hasil dapat direplikasi, penting untuk penelitian atau pengembangan produk.
- **Efisiensi Pelatihan:** Konsistensi mempercepat pelatihan model, menghemat waktu dan sumber daya komputasi.

Konsistensi dalam pengumpulan data memastikan model yang dihasilkan handal dalam pengambilan keputusan di dunia nyata. Studio box mini bisa membantu menciptakan konsistensi dalam pengambilan gambar.

b. Anotasi

Dalam pengolahan citra, anotasi adalah proses menambahkan label ke gambar untuk memberikan pemahaman tentang isinya. Anotasi membantu melatih model dalam tugas penglihatan komputer seperti deteksi objek, segmentasi, klasifikasi, atau pengenalan pola (H, 2023).

Teknik anotasi umum:

- **Bounding Box**: Menggambar kotak di sekitar objek untuk mendeteksi lokasi dan ukuran objek.

- **Segmentasi Semantik**: Memberi label pada setiap piksel dalam gambar sesuai dengan kelas objek atau area.
- **Segmentasi Instance**: Memberi label unik pada setiap objek individu dalam gambar.
- **Keypoint Annotation**: Menandai titik-titik kunci pada objek, seperti mata, hidung, dan mulut pada wajah.
- **Polygon Annotation**: Menggambar poligon mengikuti kontur objek untuk segmentasi bentuk kompleks.
- **Klasifikasi Gambar**: Memberi label kelas pada setiap gambar untuk menunjukkan isinya.

Proses anotasi citra biasanya membutuhkan waktu dan upaya yang signifikan, terutama ketika data pelatihan besar diperlukan. Konsistensi dalam anotasi dan kualitas label yang tinggi sangat penting untuk mendapatkan hasil yang baik dalam pelatihan model penglihatan computer (Raad et al., 2021).

2. Augmentasi Data

Augmentasi data meningkatkan jumlah sampel data pelatihan dengan variasi pada data yang ada. Ini bisa dilakukan dengan memutar, membalik, atau mengubah ukuran gambar, serta menambahkan noise pada data teks atau sinyal.

Tujuan utamanya adalah meningkatkan keberagaman data pelatihan agar model menjadi lebih robust, yakni mampu menghasilkan performa yang konsisten dalam berbagai kondisi (Trubin et al., 2022). Keberagaman dan kualitas data pelatihan sangat mempengaruhi keandalan model.

Dalam pra-pemrosesan gambar, beberapa teknik yang digunakan:

- **Resizing**: Mengubah ukuran gambar menjadi seragam.
- **Grayscaling**: Mengonversi gambar berwarna menjadi skala abu-abu.
- **Pengurangan Noise**: Menggunakan smoothing, blurring, dan filtering.
- **Normalisasi**: Menyesuaikan nilai intensitas piksel ke rentang yang diinginkan.
- **Binarization**: Mengonversi gambar skala abu-abu menjadi hitam-putih.
- **Peningkatan Kontras**: Menyesuaikan kontras gambar dengan histogram equalization.

Dengan teknik-teknik ini, Anda dapat meningkatkan kualitas data gambar dan membangun aplikasi visi komputer yang lebih baik (Lu et al., 2022).

a. Mengambil dan Mengkonversi Gambar dengan *Python Libraries (Loading and Converting Images with Python Libraries)*

Untuk memulai pemrosesan gambar di Python, Anda perlu memuat dan mengkonversi gambar Anda ke dalam format yang dapat dikerjakan oleh *libraries* yang digunakan. Dua pilihan yang paling populer untuk ini adalah OpenCV dan Pillow.

Memuat gambar dengan OpenCV: OpenCV dapat memuat gambar dalam format seperti PNG, JPG, TIFF, dan BMP. Anda dapat memuat gambar dengan:

```
import cv2
image = cv2.imread(path/to/image.jpg')
```

Ini akan memuat gambar sebagai larik NumPy. Gambar berada dalam ruang warna BGR, jadi Anda mungkin ingin mengkonversinya ke RGB.

Memuat gambar dengan Pillow: Pillow adalah versi PIL (*Python Image Library*) yang ramah dan mudah digunakan. Dukungannya bahkan lebih banyak daripada OpenCV, termasuk PSD, ICO, dan WEBP. Anda dapat memuat gambar dengan:

```
from PIL import Image
image = Image.open('path/to/image.jpg')
```

Gambar akan berada dalam ruang warna RGB.

b. Mengkonversi di antara ruang warna

Anda mungkin perlu mengkonversi gambar di antara ruang warna seperti RGB, BGR, HSV, dan Skala Abu-abu. Ini dapat dilakukan dengan OpenCV atau Pillow. Misalnya, untuk mengonversi BGR ke Grayscale di OpenCV, Anda dapat menggunakan:

```
image = image.convert('HSV')
```

Atau untuk mengkonversi RGB ke HSV di Pillow Anda dapat menggunakan:

```
image = image.convert('HSV')
```

c. Mengubah Ukuran (*Resizing*) dan Memotong Gambar (*Cropping*) ke Dimensi Standar

Mengubah ukuran dan memotong gambar adalah langkah penting dalam pra-pemrosesan. Algoritma pembelajaran mesin memerlukan ukuran standar, seringkali 224x224 atau 256x256 piksel. Pada Python, Anda bisa menggunakan OpenCV atau Pillow. Dengan OpenCV, Anda dapat menggunakan fungsi resize() untuk mengubah ukuran dan crop() untuk memotong menjadi persegi. Dengan Pillow, Anda dapat menggunakan Image.open() dan resize(). Langkah ini penting karena memungkinkan model memproses gambar lebih efisien dan dapat meningkatkan akurasi (Demant et al., 2013).

3. Menangani Kumpulan Data yang Tidak Seimbang (Imbalanced Datasets)

Ketidakseimbangan dataset terjadi ketika jumlah sampel dalam satu kelas jauh lebih banyak daripada yang lain. Ini menjadi masalah dalam pembelajaran mesin karena model cenderung memihak kelas mayoritas, menghasilkan bias terhadap kelas minoritas.

Beberapa pendekatan untuk menangani ketidakseimbangan dataset:

1. **Oversampling**: Menambah sampel dari kelas minoritas untuk menyamakan jumlah sampel di setiap kelas. Namun, oversampling acak dapat menyebabkan overfitting (Thumpati & Zhang, 2023).
2. **Undersampling**: Mengurangi sampel dari kelas mayoritas untuk menyamakan jumlah sampel. Ini mendorong model memperhatikan kelas minoritas,

tetapi bisa mengorbankan informasi penting dari kelas mayoritas (Kusdiyanto & Pristyanto, 2022).

3. **Pembobotan Kelas**: Memberikan bobot lebih tinggi pada kelas minoritas saat melatih model. Ini memaksa model lebih memperhatikan kelas minoritas selama pembelajaran (Budania et al., 2020).

Selain itu, teknik resampling adaptif, pembelajaran transfer, atau algoritma khusus dapat digunakan. Mempertimbangkan keuntungan dan kerugian setiap pendekatan serta karakteristik data spesifik adalah penting. Kombinasi beberapa pendekatan kadang memberikan hasil terbaik.

4. *Dimensionality Reduction* (*Feature Selection*) untuk CV

Dalam computer vision, dimensi data merujuk pada jumlah fitur atau atribut yang digunakan untuk mewakili setiap gambar. Setiap piksel dalam gambar dapat dianggap sebagai fitur, dan gambar berwarna seringkali memiliki tiga kanal warna (misalnya, merah, hijau, dan biru), sehingga setiap piksel memiliki tiga nilai warna. Ini berarti gambar berwarna dapat direpresentasikan dalam ruang fitur tiga dimensi.

Namun, dalam kenyataannya, gambar-gambar sering memiliki resolusi yang tinggi dan jumlah piksel yang besar, yang menghasilkan ruang fitur dengan dimensi yang sangat tinggi. Misalnya, gambar berukuran 1000 x 1000 piksel akan memiliki satu juta fitur jika setiap piksel direpresentasikan secara terpisah. Ini membuat komputasi dan analisis menjadi sangat mahal.

Dimensionality reduction adalah proses mengurangi jumlah fitur tersebut menjadi jumlah yang lebih kecil, tetapi tetap mempertahankan sebanyak mungkin informasi yang

berguna. Ini dapat dilakukan dengan beberapa teknik seperti PCA, LDA, t-SNE, atau autoencoders.

Dalam computer vision, terdapat beberapa teknik reduksi dimensionalitas yang umum digunakan:

1. ***Principal Component Analysis*** **(PCA):** PCA adalah teknik yang umum digunakan untuk mengurangi dimensionalitas data. Ini mentransformasikan data ke ruang fitur yang lebih rendah berdasarkan variansi terbesar. Dalam citra, PCA dapat digunakan untuk mengurangi dimensi dari setiap piksel ke komponen utama yang mewakili variasi dalam dataset (Gautam & Dey, 2022).
2. ***Linear Discriminant Analysis*** **(LDA):** LDA mirip dengan PCA, tetapi mempertimbangkan label kelas dalam proses reduksi dimensionalitas. Ini bertujuan untuk mempertahankan informasi yang paling relevan untuk membedakan antara kelas-kelas yang berbeda. Dalam computer vision, LDA membantu mempertahankan informasi yang paling diskriminatif antara kelas objek (Koziarski, 2021).
3. ***t-Distributed Stochastic Neighbor Embedding*** **(t-SNE):** t-SNE adalah teknik non-linear untuk visualisasi data di ruang dimensi rendah. Ini mempertahankan struktur jarak antara titik data dalam ruang dimensi tinggi saat meletakkannya ke ruang dimensi rendah. Dalam computer vision, t-SNE dapat digunakan untuk visualisasi data atau ekstraksi fitur non-linear (Cicak & Avci, 2023).
4. ***Autoencoders:*** Autoencoder adalah jaringan syaraf tiruan untuk belajar representasi data yang kompres. Ini terdiri dari encoder yang mengkonversi input ke ruang dimensi rendah dan decoder yang merekonstruksi input dari representasi tersebut. Dalam computer vision, autoencoders mempelajari representasi fitur yang lebih ringkas dari gambar (Rezvani & Wang, 2021)

DAFTAR PUSTAKA

Budania, S., Kumar, T., Kumar, H., & Nikam, G. (2020). Hybrid Machine Intelligence for Imbalanced Data. *Social Science Research Network*. https://doi.org/10.2139/ssrn.3602531

Cicak, S., & Avci, U. (2023). Handling Imbalanced Data in Predictive Maintenance: A Resampling-Based Approach. *2023 5th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 1–6. https://doi.org/10.1109/HORA58378.2023.10156799

Demant, C., Garnica, C., & Streicher-Abel, B. (2013). *Overview: Image Preprocessing*. 25–63. https://doi.org/10.1007/978-3-642-33905-9_2

Gautam, S., & Dey, R. (2022). METHODS FOR CLASSIFICATION OF IMBALANCED DATA: A REVIEW. *International Research Journal of Computer Science*. https://doi.org/10.26562/irjcs.2021.v0904.004

H, J. D. K. (2023). Implementation and Efficient Analysis of Preprocessing Techniques in Deep Learning for Image Classification. *Current Medical Imaging*. https://doi.org/10.2174/1573405620666230829150157

Koziarski, M. (2021). Potential Anchoring for imbalanced data classification. *Pattern Recognit.*, *120*, 108114. https://doi.org/10.1016/j.patcog.2021.108114

Kusdiyanto, A. Y., & Pristyanto, Y. (2022). Machine Learning Models for Classifying Imbalanced Class Datasets Using Ensemble Learning. *2022 5th International Seminar on Research of Information Technology and Intelligent*

Systems (ISRITI), 648–653. https://doi.org/10.1109/ISRITI56927.2022.10052887

Lu, G., Ge, X., Zhong, T., Geng, J., & Hu, Q. (2022). Preprocessing Enhanced Image Compression for Machine Vision. *ArXiv*, *abs/2206.05650*. https://doi.org/10.48550/arXiv.2206.05650

Minh, T., Sinn, M., Lam, H. T., & Wistuba, M. (2018). Automated Image Data Preprocessing with Deep Reinforcement Learning. *ArXiv*, *abs/1806.05886*.

Raad, K. B. D., van Garderen, K. A., Smits, M., Voort, S. V. D., Incekara, F., Oei, E., Hirvasniemi, J., Klein, S., & Starmans, M. P. (2021). The Effect of Preprocessing on Convolutional Neural Networks for Medical Image Segmentation. *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, 655–658. https://doi.org/10.1109/ISBI48211.2021.9433952

Rezvani, S., & Wang, X. (2021). Class imbalance learningusing fuzzy ART and intuitionistic fuzzy twin support vector machines. *Inf. Sci.*, *578*, 659–682. https://doi.org/10.1016/J.INS.2021.07.010

Stojnev, D., & Ilić, A. (2020). *Preprocessing Image Data for Deep Learning*. https://doi.org/10.15308/sinteza-2020-312-317

Thumpati, A., & Zhang, Y. (2023). Towards Optimizing Performance of Machine Learning Algorithms on Unbalanced Dataset. *Artificial Intelligence & Applications*. https://doi.org/10.5121/csit.2023.131914

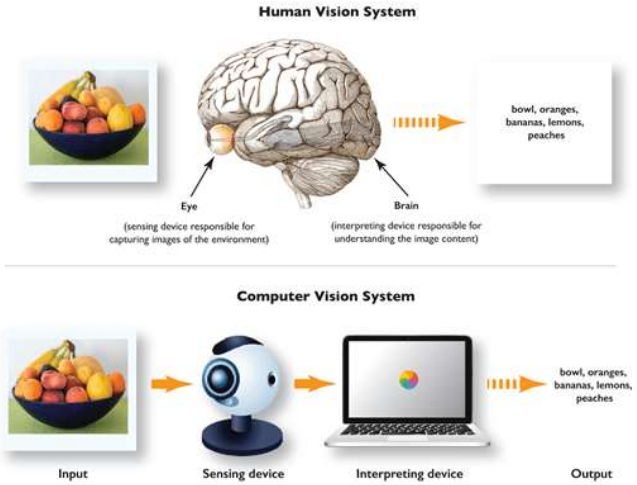
Trubin, A. E., Morozov, A., Zubanova, A. E., Ozheredov, V., & Korepanova, V. (2022). The method of preprocessing machine learning data for solving computer vision problems. *Journal Of Applied Informatics*. https://doi.org/10.37791/2687-0649-2022-17-4-47-56

BAB 6
COMPUTER VISION DENGAN CONVOLUTIONAL NEURAL NETWORKS (CNNS)

Oleh Arvita Agus Kurniasari, S.ST.,M.Tr.Kom

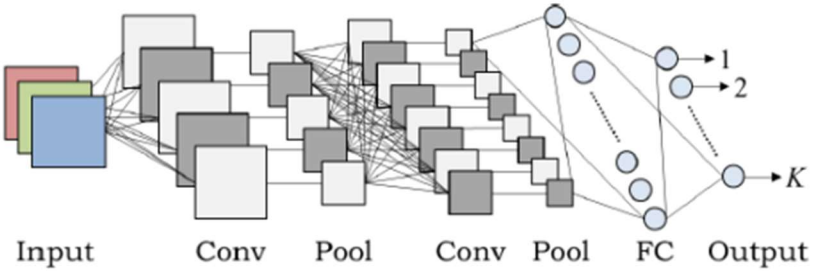
1. Pendahuluan

Convolutional Neural Networks (CNNs) merupakan bagian dari pemodelan *deep learning* yang mendapatkan popularitas signifikan karena performanya yang luar biasa di berbagai bidang, khususnya dalam *Computer Vision*. CNNs dirancang untuk meniru kemampuan pemrosesan visual otak manusia dengan menggunakan lapisan konvolusi untuk mengekstraksi fitur dari data *input* (Krizhevsky et al., 2017). Struktur CNNs berisi dua bagian tahapan yaitu tahap ekstraksi fitur dan proses klasifikasi. Proses ekstraksi pada CNNs terjadi pada lapisan konvolusi dan *pooling*, yang membentuk peta fitur (*feature maps*). Di CNNs, *output* lapisan konvolusi pertama digunakan sebagai *input* untuk lapisan berikutnya, sehingga CNNs berfungsi secara hierarkis. Proses klasifikasi pada CNNs *output* dari proses ekstraksi fitur diubah ke dalam satu dimensi kemudian dimasukkan ke dalam pengklasifikasi sebagai *fully-connected layer* atau *classifier* (Miftakhurrokhmat et al., 2021).



Gambar 12. *Convolutional Neural Network* (CNNs)

Variasi arsitektur *Convolutional Neural Network* (CNNs) yang berbeda-beda terdiri dari beberapa lapisan yang berfungsi secara spesifik untuk memproses dan menganalisis data visual (Lusiana et al., 2024). Lapisan-lapisan tersebut meliputi:



Gambar 13. Arsitektur *Convolutional Neural Network* (CNNs) Arsitektur CNN (Adiatma et al., 2021)

1. ***Input Layer***: Lapisan ini menerima input berupa citra atau data visual yang akan diproses oleh CNNs.
2. ***Convolutional Layer***: Lapisan ini melakukan proses konvolusi pada input, yaitu mengalikan setiap piksel dengan filter yang telah dipelajari dan menghitung hasilnya. Proses ini memungkinkan CNNs untuk mendeteksi pola dan fitur yang tersembunyi dalam citra.
3. ***ReLU Layer***: Lapisan ini melakukan aktivasi fungsi ReLU (*Rectified Linear Unit*) pada hasil konvolusi. Fungsi ReLU mengaktifkan nilai yang lebih besar dari nol dan menghapus nilai negatif, sehingga memungkinkan CNNs untuk mempertahankan informasi yang relevan.
4. ***Pooling Layer***: Lapisan ini melakukan proses pooling, yaitu mengurangi ukuran citra dengan menghitung nilai rata-rata atau nilai maksimum dari setiap blok piksel. Proses ini memungkinkan CNNs untuk mengurangi kompleksitas citra dan mempercepat proses pengolahan.
5. ***Flatten Layer***: Lapisan ini mengubah struktur citra yang telah dipooling menjadi vektor yang lebih sederhana, memungkinkan CNNs untuk memproses citra secara lebih efektif.
6. ***Fully Connected Layer***: Lapisan ini melakukan klasifikasi atau prediksi berdasarkan vektor yang telah diproses. Lapisan ini terdiri dari neuron yang saling terhubung dan memungkinkan CNNs untuk membuat keputusan berdasarkan informasi yang telah diproses.
7. ***Softmax Layer***: Lapisan ini melakukan transformasi *softmax* pada hasil klasifikasi, memungkinkan CNNs untuk menghasilkan probabilitas yang lebih akurat dan memprediksi kelas yang paling mungkin.

2. Studi Kasus Convolutional Neural Network (CNNs)

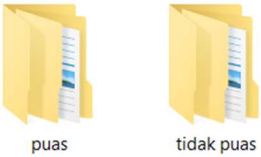
a. Data

Pada implementasi Metode CNNs ini dilakukan dengan menggunakan dataset dari sumber terbuka, yaitu https://www.kaggle.com/datasets/msambare/fer2013 (FER-2013) yang tersedia di Kaggle. Dataset ini berisi gambar wajah yang telah diklasifikasikan berdasarkan ekspresi emosional menjadi beberapa kategori: *angry*, *disgust*, *fear*, *happy*, *sad*, *surprise,* dan *neutral*. Untuk keperluan penelitian ini, kategori-kategori tersebut kemudian dikelompokkan menjadi dua kategori utama, yaitu puas dan tidak puas.
Kategori puas mencakup dua ekspresi: *happy* dan *neutral*. Ekspresi happy mencerminkan keadaan emosional yang positif dan jelas menunjukkan rasa kepuasan. Sementara itu, ekspresi neutral dipandang sebagai keadaan emosional yang netral atau tidak menunjukkan ketidakpuasan, sehingga dimasukkan ke dalam kategori puas.

Sebaliknya, kategori tidak puas terdiri dari lima ekspresi: *angry*, *disgust*, *fear*, *sad*, dan *surprise*. Ekspresi *angry* (marah) dan *disgust* (jijik) jelas mencerminkan ketidakpuasan yang kuat. Ekspresi *fear* (takut) dan *sad* (sedih) juga mencerminkan keadaan emosional negatif yang umumnya berkaitan dengan ketidakpuasan. Sementara itu, ekspresi *surprise* (terkejut) dapat dianggap sebagai reaksi terhadap situasi yang tidak terduga, yang seringkali berkaitan dengan ketidakpuasan.

Setelah melakukan pengelompokan, data dari masing-masing kategori dianalisis untuk memahami distribusi serta karakteristik dari setiap ekspresi dalam konteks kategori puas dan tidak puas. Pengelompokan ini tidak hanya bertujuan untuk menyederhanakan analisis data tetapi juga untuk

mengeksplorasi perbedaan antara emosi positif/netral dan emosi negatif dalam kaitannya dengan kepuasan atau ketidakpuasan. Dengan demikian, diharapkan dapat memberikan wawasan yang lebih mendalam mengenai bagaimana ekspresi wajah dapat digunakan sebagai indikator kepuasan atau ketidakpuasan dalam berbagai konteks.



Gambar 14. Kategori Model

Data training terdiri dari 4000 citra, dengan 2000 citra dikategorikan sebagai puas dan 2000 citra dikategorikan sebagai tidak puas. Data ini digunakan untuk melatih model sehingga dapat mengenali pola-pola yang mengindikasikan kepuasan atau ketidakpuasan dari ekspresi wajah.

Data testing terdiri dari 1000 citra, dengan 500 citra dikategorikan sebagai puas dan 500 citra dikategorikan sebagai tidak puas. Data testing ini digunakan untuk mengevaluasi kinerja model yang telah dilatih, dengan menguji seberapa baik model tersebut dapat mengklasifikasikan citra baru yang tidak pernah dilihat sebelumnya ke dalam kategori puas atau tidak puas.

Tabel 1. Kategori Model

Dataset	Jumlah total	puas	Tidak puas
training	4000	2000	2000
testing	1000	500	500

b. Implementasi Python

Berikut adalah implementasi metode Convolutional Neural Networks (CNNs) menggunakan Python di Google Colab. Implementasi ini mencakup persiapan data, pembuatan model CNN, pelatihan model, dan evaluasi kinerja model. Data yang digunakan berasal dari dataset FER-2013 yang sebelumnya telah diunduh dan dipasang ke dalam Google Colab dari Google Drive.

1. Persiapan Lingkungan dan Data

Persiapan lingkungan dan data adalah tahap penting dalam siklus analisis data dan pemodelan machine learning. Tahap ini memastikan bahwa data yang digunakan dalam proses pelatihan model berada dalam format yang tepat dan dapat diakses dengan mudah.

```
from google.colab import drive
drive.mount('/content/drive/')
```

```
training = '/content/drive/My Drive/Buku/2024/computer vision/dataset/train'
test = '/content/drive/My Drive/Buku/2024/computer vision/dataset/test'
```

```
import cv2
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Dense, Dropout, Flatten
from keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import numpy as np
from keras.regularizers import l2
from sklearn.metrics import confusion_matrix, classification_report, roc_auc_score, f1_score, roc_curve, auc
import seaborn as sns
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
```

Kode sumber di atas mengimplementasikan sebuah model jaringan saraf convolutional (Convolutional Neural Network atau CNN) menggunakan Keras, untuk tujuan klasifikasi gambar. Berikut adalah penjelasan dari fungsi dan bagian-bagian dari kode *Import Library* tersebut:

a) **cv2**: OpenCV digunakan untuk pemrosesan gambar.
b) **keras.models.Sequential**: Digunakan untuk membangun model sekuensial.
c) **keras.layers.Conv2D, MaxPooling2D, Dense, Dropout, Flatten**: Layer-layer yang digunakan untuk membentuk CNN.
d) **keras.optimizers.Adam**: Optimizer Adam digunakan untuk proses training model.
e) **keras.preprocessing.image.ImageDataGenerator**: Untuk augmentasi gambar selama proses training.
f) **matplotlib.pyplot**: Untuk visualisasi data dan hasil.
g) **numpy**: Untuk operasi matematika pada array.
h) **keras.regularizers.l2**: Digunakan untuk regularisasi L2.
i) **sklearn.metrics**: Digunakan untuk evaluasi model.
j) **seaborn**: Untuk visualisasi data dengan lebih baik.
k) **keras.callbacks.ModelCheckpoint, EarlyStopping, ReduceLROnPlateau**: Callbacks untuk mengontrol dan menyimpan proses training.

2. Pengolahan Data

Pengolahan data adalah langkah penting dalam proses analisis data yang bertujuan untuk mempersiapkan data mentah agar siap digunakan dalam model pembelajaran mesin.

```
train_data_gen = ImageDataGenerator(rescale=1./255)
validation_data_gen = ImageDataGenerator(rescale=1./255)

train_generator = train_data_gen.flow_from_directory(
    training,
    target_size=(48, 48),
    batch_size=32,
    color_mode="grayscale",
    class_mode='categorical',
    shuffle=True
)

validation_generator = validation_data_gen.flow_from_directory(
    test,
    target_size=(48, 48),
    batch_size=32,
    color_mode="grayscale",
    class_mode='categorical',
    shuffle=False
)
```

Kode sumber di atas berfungsi untuk mempersiapkan data latih dan data validasi dengan menggunakan ImageDataGenerator dari Keras, yang akan mengalirkan data gambar secara batch selama proses training model jaringan saraf convolutional (CNN). Berikut adalah penjelasan dari fungsi dan bagian-bagian dari kode tersebut:

a) ImageDataGenerator:
   Rescale: Parameter rescale=1./255 digunakan untuk menormalisasi nilai piksel gambar dari rentang [0, 255] ke rentang [0, 1]. Normalisasi ini penting untuk mempercepat konvergensi model selama proses training.

b) Flow from Directory:
   1. train_generator dan validation_generator: flow_from_directory adalah metode yang digunakan untuk menghasilkan data gambar secara batch dari

direktori yang terstruktur sesuai dengan label klasifikasi.
2. training dan test adalah direktori yang berisi sub-direktori dengan nama kelas untuk data latih dan data uji.

C) Parameter yang digunakan:
1. **target_size=(48, 48)**: Mengubah ukuran semua gambar menjadi 48x48 piksel. Ukuran ini harus disesuaikan dengan ukuran input yang diharapkan oleh model.
2. **batch_size=32**: Menentukan jumlah gambar yang akan diproses dalam satu batch.
3. **color_mode="grayscale"**: Menentukan bahwa gambar akan dikonversi ke skala abu-abu. Ini berarti gambar hanya memiliki satu channel warna.
4. **class_mode='categorical'**: Menentukan bahwa label yang dihasilkan adalah one-hot encoded, yang cocok untuk masalah klasifikasi multi-kelas.
5. **shuffle=True (untuk train_generator)**: Mengacak urutan data gambar untuk memastikan bahwa model tidak belajar urutan tertentu, yang bisa menyebabkan overfitting.
6. **shuffle=False (untuk validation_generator)**: Tidak mengacak data validasi untuk memastikan evaluasi yang konsisten.

3. Pembuatan Model CNN

Pembuatan model Convolutional Neural Network (CNN) adalah proses mendesain dan menginisialisasi arsitektur jaringan saraf tiruan yang digunakan untuk tugas pengenalan pola, dalam hal ini klasifikasi ekspresi wajah.

```
emotion_model = Sequential()

#layer 1
emotion_model.add(Conv2D(32, kernel_size=(3, 3),padding='same',activation='relu',
                         input_shape=(48, 48, 1)))
emotion_model.add(Conv2D(64, kernel_size=(3, 3),padding='same', activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.2))

#layer 2
emotion_model.add(Conv2D(128, kernel_size=(3, 3),padding='same', activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.2))

#layer 3
emotion_model.add(Conv2D(128, kernel_size=(3, 3),padding='same', activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.2))

emotion_model.add(Flatten())
emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.2))
emotion_model.add(Dense(2, activation='softmax'))
```

Kode sumber di atas mendefinisikan dan mengkonfigurasi model jaringan saraf convolutional (Convolutional Neural Network atau CNN) menggunakan Keras, yang digunakan untuk klasifikasi emosi berdasarkan gambar. Berikut adalah penjelasan dari fungsi dan bagian-bagian dari kode tersebut:

a) Inisialisasi Model:
   emotion_model = Sequential(): Mendefinisikan model sekuensial yang berarti lapisan-lapisan model akan ditambahkan secara berurutan.

b) Layer 1:
   emotion_model.add(Conv2D(32, kernel_size=(3, 3), padding='same', activation='relu', input_shape=(48, 48, 1)))

Menambahkan layer konvolusi dengan 32 filter, masing-masing berukuran 3x3.

1. padding='same' memastikan ukuran output sama dengan input dengan menambahkan padding di sekitar input.
2. activation='relu' menggunakan fungsi aktivasi ReLU (Rectified Linear Unit) untuk menambahkan non-linearitas ke model.

3. input_shape=(48, 48, 1) menunjukkan ukuran input gambar yang berukuran 48x48 piksel dengan satu channel (grayscale).

   emotion_model.add(Conv2D(64, kernel_size=(3, 3), padding='same', activation='relu'))

   Menambahkan layer konvolusi kedua dengan 64 filter, masing-masing berukuran 3x3.

   emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

   Menambahkan layer pooling dengan ukuran pool 2x2 untuk melakukan down-sampling, yang mengurangi dimensi spasial (tinggi dan lebar) dari output.

   emotion_model.add(Dropout(0.2))

   Menambahkan dropout layer dengan probabilitas 0.2 untuk mencegah overfitting dengan mengabaikan 20% dari neuron secara acak selama training.

c) Layer 2

   emotion_model.add(Conv2D(128, kernel_size=(3, 3), padding='same', activation='relu'))

   Menambahkan layer konvolusi dengan 128 filter, masing-masing berukuran 3x3.

   emotion_model.add(MaxPooling2D(pool_size=(2, 2)))

   Menambahkan layer pooling dengan ukuran pool 2x2 untuk melakukan down-sampling.

emotion_model.add(Dropout(0.2))
Menambahkan dropout layer dengan probabilitas 0.2.

d) Layer 3
emotion_model.add(Conv2D(128, kernel_size=(3, 3), padding='same', activation='relu'))
Menambahkan layer konvolusi dengan 128 filter, masing-masing berukuran 3x3.
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
Menambahkan layer pooling dengan ukuran pool 2x2 untuk melakukan down-sampling.
emotion_model.add(Dropout(0.2))
Menambahkan dropout layer dengan probabilitas 0.2.

e) Fully Connected Layers
emotion_model.add(Flatten())
Mengubah output dari layer konvolusi terakhir yang berbentuk matriks menjadi vektor 1D, sehingga dapat dihubungkan ke layer dense.
emotion_model.add(Dense(1024, activation='relu'))
Menambahkan layer dense (fully connected) dengan 1024 neuron dan fungsi aktivasi ReLU.
emotion_model.add(Dropout(0.2))

Menambahkan dropout layer dengan probabilitas 0.2.
emotion_model.add(Dense(2, activation='softmax'))
Menambahkan layer dense terakhir dengan 2 neuron dan fungsi aktivasi softmax, yang mengubah output menjadi probabilitas untuk klasifikasi dua kelas (misalnya, dua jenis emosi).

4. Pelatihan Model

Pelatihan model adalah proses di mana model machine learning disesuaikan dengan data training untuk mempelajari pola-pola yang ada dalam data tersebut. Tahapan dalam pelatihan model CNN melibatkan serangkaian langkah yang mendetail untuk menyesuaikan bobot dan bias dalam jaringan saraf tiruan.

```
cv2.ocl.setUseOpenCL(False)
 emotion_model.compile(
     loss='categorical_crossentropy',
     optimizer = Adam(learning_rate=0.001),
     metrics=['accuracy'])
checkpoint = ModelCheckpoint('model_weights.h5', monitor='accuracy',
                             verbose=1, save_best_only=True)

ealy_stopping = EarlyStopping(monitor='accuracy',
                              min_delta=0,
                              patience=3,
                              verbose=1,
                              restore_best_weights=True)

reduce_learningrate = ReduceLROnPlateau(monitor='accuracy',
                                        factor=0.2,
                                        patience=3,
                                        verbose=1,
                                        min_delta=0.0001)
callbacks_list = [checkpoint]
emotion_model_info = emotion_model.fit(
        train_generator,
        steps_per_epoch=4000  // 32,
        epochs=50,
        validation_data=validation_generator,
        validation_steps=1000 // 32,
        callbacks=callbacks_list)
```

kode di atas adalah bagian dari proses pelatihan model menggunakan Keras, yang mencakup beberapa tahapan dan

pengaturan yang diperlukan untuk mengoptimalkan kinerja model. Berikut adalah penjelasan dari setiap bagian kode tersebut:

1. **cv2.ocl.setUseOpenCL(False)**
   Menonaktifkan penggunaan OpenCL oleh OpenCV.
2. **emotion_model.compile(...)**
   Mengompilasi model dengan menentukan loss function, optimizer, dan metrik evaluasi.
3. **checkpoint = ModelCheckpoint(...)**
   Membuat objek callback ModelCheckpoint untuk menyimpan bobot model saat kinerja terbaik tercapai selama pelatihan.
4. **early_stopping = EarlyStopping(...)**
   Membuat objek callback EarlyStopping untuk menghentikan pelatihan jika tidak ada peningkatan dalam metrik yang
5. **reduce_learningrate = ReduceLROnPlateau(...)**
   Membuat objek callback ReduceLROnPlateau untuk mengurangi laju pembelajaran (learning rate) jika tidak ada peningkatan dalam metrik yang diamati.
6. **callbacks_list = [checkpoint]**
   Membuat daftar callbacks yang akan digunakan selama proses pelatihan model.
7. **emotion_model_info = emotion_model.fit(...)**
   Memulai proses pelatihan model dengan data training dan data validation yang disediakan. Hasil pelatihan akan disimpan dalam variabel emotion_model_info untuk analisis lebih lanjut.

Proses pelatihan selama 50 *epoch* menunjukkan peningkatan yang signifikan dalam akurasi, dari sekitar 48,9% menjadi hampir 98,7% pada set pelatihan. Namun, akurasi validasi menunjukkan tanda-tanda mencapai plateau dan bahkan

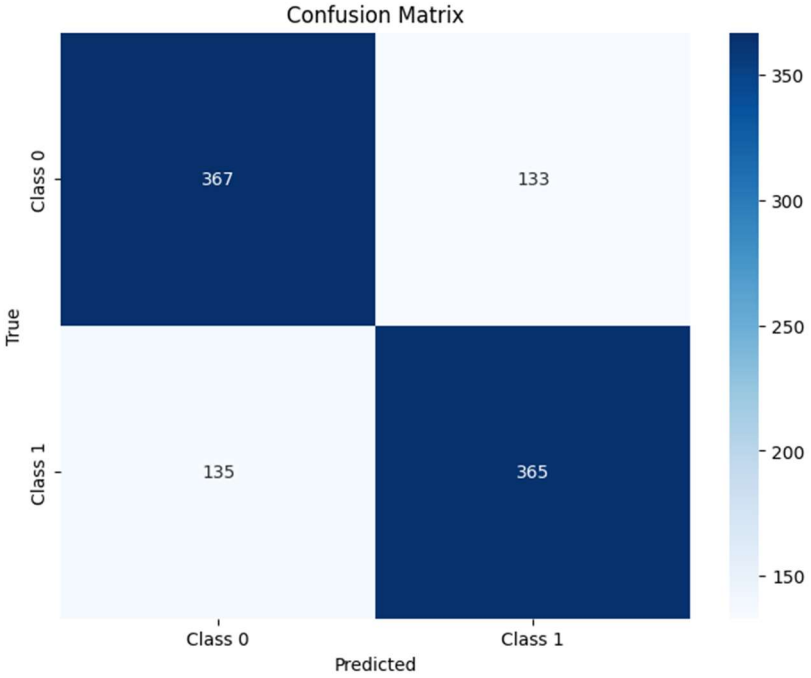
sedikit menurun pada beberapa *epoch*, yang mengindikasikan adanya potensi *overfitting*.

```
Epoch 1/50
125/125 [==============================] - ETA: 0s - loss: 0.7059 - accuracy: 0.4893
Epoch 1: accuracy improved from -inf to 0.48925, saving model to model_weights.h5
125/125 [==============================] - 921s 7s/step - loss: 0.7059 - accuracy: 0.4893 - val_loss: 0.6929 - val_accuracy: 0.4970
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `
  saving_api.save_model(
Epoch 2/50
125/125 [==============================] - ETA: 0s - loss: 0.6892 - accuracy: 0.5487
Epoch 2: accuracy improved from 0.48925 to 0.54875, saving model to model_weights.h5
125/125 [==============================] - 104s 830ms/step - loss: 0.6892 - accuracy: 0.5487 - val_loss: 0.6883 - val_accuracy: 0.5373
Epoch 3/50
125/125 [==============================] - ETA: 0s - loss: 0.6633 - accuracy: 0.6018
Epoch 3: accuracy improved from 0.54875 to 0.60175, saving model to model_weights.h5
125/125 [==============================] - 106s 850ms/step - loss: 0.6633 - accuracy: 0.6018 - val_loss: 0.6555 - val_accuracy: 0.6240
Epoch 4/50
125/125 [==============================] - ETA: 0s - loss: 0.6452 - accuracy: 0.6205
Epoch 48/50
125/125 [==============================] - ETA: 0s - loss: 0.0416 - accuracy: 0.9850
Epoch 48: accuracy did not improve from 0.98900
125/125 [==============================] - 105s 840ms/step - loss: 0.0416 - accuracy: 0.9850 - val_loss: 1.2219 - val_accuracy: 0.7560
Epoch 49/50
125/125 [==============================] - ETA: 0s - loss: 0.0408 - accuracy: 0.9870
Epoch 49: accuracy did not improve from 0.98900
125/125 [==============================] - 104s 830ms/step - loss: 0.0408 - accuracy: 0.9870 - val_loss: 1.2710 - val_accuracy: 0.7379
Epoch 50/50
125/125 [==============================] - ETA: 0s - loss: 0.0415 - accuracy: 0.9862
Epoch 50: accuracy did not improve from 0.98900
125/125 [==============================] - 105s 842ms/step - loss: 0.0415 - accuracy: 0.9862 - val_loss: 1.3419 - val_accuracy: 0.7308
```

5. Visualisasi Matriks

Confusion Matrix adalah alat evaluasi yang sangat berguna dalam analisis kinerja model klasifikasi, terutama dalam konteks model pembelajaran mesin dan jaringan saraf. Dengan matriks ini, dapat memahami distribusi kesalahan model dan mengidentifikasi pola prediksi yang salah, yang pada gilirannya dapat membantu dalam pengembangan model yang lebih baik dan penyetelan hyperparameter yang lebih efektif.

```
conf_matrix = confusion_matrix(y_true, y_pred_labels)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Class 0', 'Class 1'],
            yticklabels=['Class 0', 'Class 1'])
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.savefig('Confusion Matrix 1.png')
plt.show()
```



6. Evaluasi Model Klasifikasi

Evaluasi model klasifikasi adalah langkah penting untuk memahami kinerja model dan memastikan bahwa model dapat membuat prediksi yang akurat dan dapat diandalkan.

```
class_report = classification_report(y_true, y_pred_labels,
                                     target_names=['Class 0', 'Class 1'])
print(class_report)

# Calculate and print AUC
roc_auc = roc_auc_score(y_true, y_pred_labels)
print(f'AUC: {roc_auc}')

# Calculate and print F1 score
f1 = f1_score(y_true, y_pred_labels)
print(f'F1 Score: {f1}')
```

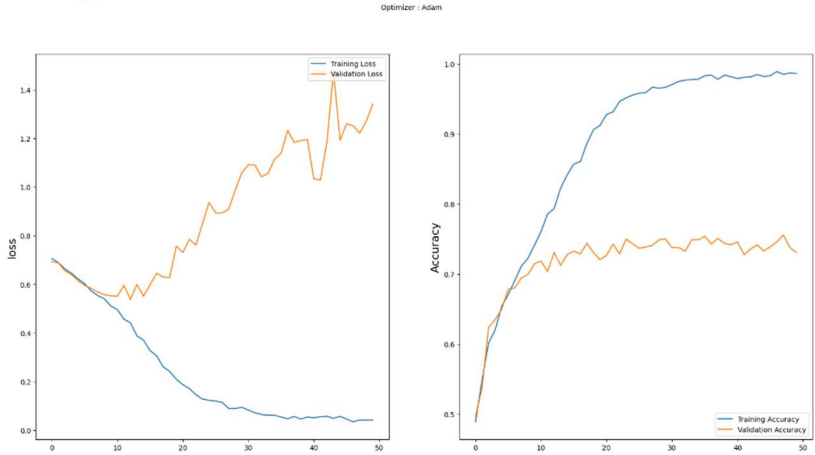
```
              precision    recall  f1-score   support

     Class 0       0.73      0.73      0.73       500
     Class 1       0.73      0.73      0.73       500

    accuracy                           0.73      1000
   macro avg       0.73      0.73      0.73      1000
weighted avg       0.73      0.73      0.73      1000

AUC: 0.732
F1 Score: 0.7314629258517035
```

Model klasifikasi menunjukkan performa yang baik dengan akurasi, presisi, recall, dan F1-Score yang seimbang di kedua kelas. Nilai AUC yang mencapai 0.732 juga menunjukkan bahwa model memiliki kemampuan yang cukup baik dalam membedakan antara kelas positif dan negatif. Namun, masih ada ruang untuk perbaikan, terutama dalam meningkatkan AUC dan F1-Score.

Untuk meningkatkan performa model, beberapa langkah yang dapat diambil meliputi:

1) Peningkatan Data Training: Mengumpulkan lebih banyak data atau melakukan augmentasi data untuk meningkatkan kualitas dan kuantitas data pelatihan.
2) Tuning Hyperparameter: Mengoptimalkan hyperparameter model untuk menemukan setelan yang menghasilkan performa terbaik.
3) Mencoba Arsitektur Model yang Berbeda: Menggunakan arsitektur model yang berbeda atau lebih kompleks yang mungkin lebih cocok untuk dataset yang digunakan.

```
plt.figure(figsize=(20,10))
plt.subplot(1, 2, 1)
plt.suptitle('Optimizer : Adam', fontsize=10)
plt.ylabel('loss', fontsize=16)
plt.plot(emotion_model_info.history['loss'], label='Training Loss')
plt.plot(emotion_model_info.history['val_loss'], label='Validation Loss')
plt.legend(loc='upper right')

plt.subplot(1, 2, 2)
plt.ylabel('Accuracy', fontsize=16)
plt.plot(emotion_model_info.history['accuracy'], label='Training Accuracy')
plt.plot(emotion_model_info.history['val_accuracy'],
         label='Validation Accuracy')
plt.legend(loc='lower right')
plt.savefig('Accuracy 1.png')
plt.show()
```



DAFTAR PUSTAKA

Adiatma, B. C. L., Utami, E., & Hartanto, A. D. (2021). Pengenalan Ekspresi Wajah Menggunakan Deep Convolutional Neural Network. *Explore*, *11*(2), 75. https://doi.org/10.35200/explore.v11i2.478

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90.

Lusiana, Kurniasari, A. A., & Kusumawati, I. F. (2024). Enhancing Accuracy in the Detection of Pneumonia in Adult Patients: An Approach by Using Convolutional Neural Networks. In T. Triwiyanto, A. Rizal, & W. Caesarendra (Ed.), *Proceedings of the 4th International Conference on Electronics, Biomedical Engineering, and Health Informatics* (hal. 593–609). Springer Nature Singapore.

Miftakhurrokhmat, Rajagede, R. A., & Rahmadi, R. (2021). Presensi Kelas Berbasis Pola Wajah, Senyum dan Wi-Fi Terdekat dengan Deep Learning. *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, *5*(1), 31–38. https://doi.org/10.29207/resti.v5i1.2575

BAB 7
COMPUTER VISION DENGAN SINGLE SHOT MULTIBOX DETECTOR (SSD)

Oleh Muhammad Hafidh Firmansyah, S.Tr.Kom., M.Sc.

1. Pendahuluan

Salah satu komponen utama kecerdasan buatan adalah deteksi objek, yang berguna untuk berbagai aplikasi, seperti pengawasan keamanan dan kendaraan otonom. Kemajuan teknologi telah mendorong pengembangan teknik deteksi objek yang semakin canggih dan efektif. Single Shot Multibox Detector (SSD) adalah salah satu teknik yang telah menonjol di bidang ini. SSD adalah pendekatan deteksi objek yang memungkinkan untuk mendeteksi objek dalam gambar dengan cepat dan akurat.

Pemahaman SSD Multibox dan kemampuan untuk menerapkannya dengan Python sangat penting bagi peneliti dan praktisi kecerdasan buatan serta pengembang yang ingin mengembangkan solusi deteksi objek yang inovatif. memahami konsep dan metode yang mendasari SSD Multibox. Penulis berharap pembaca mendapatkan pemahaman yang mendalam tentang SSD Multibox dan bagaimana menggunakannya untuk memenuhi kebutuhan proyek mereka. Selain itu, kami akan membahas beberapa studi kasus tentang penerapan SSD dalam berbagai bidang, memberikan wawasan bermanfaat tentang bagaimana teknologi deteksi objek ini dapat diterapkan dalam situasi dunia nyata.

Dengan mendapatkan pengetahuan dan kemampuan yang diperlukan dari buku ini, pembaca diharapkan dapat mengatasi tantangan saat ini dalam mendeteksi objek dengan lebih percaya diri dan mengambil langkah-langkah menuju kemajuan dalam bidang ini. Selamat membaca, dan semoga buku ini membantu Anda belajar deteksi objek dengan SSD Multibox berbasis Python.

2. Dasar Teori

Bab ini akan membahas konsep dasar yang menjadi landasan pemahaman deteksi objek dan memberikan tinjauan singkat tentang algoritma Deep Learning yang digunakan dalam SSD. Kita juga akan mempelajari jaringan konvolusi dan bagaimana pentingnya untuk pengolahan gambar.

- **Konsep dasar tentang deteksi objek.**

Salah satu bagian penting dari pengolahan citra adalah deteksi objek(Anushka et al., 2021; Kaur and Singh, 2022), yang bertujuan untuk menemukan keberadaan dan lokasi objek yang menarik dalam gambar. Proses ini melibatkan beberapa langkah, seperti mengekstraksi fitur dari objek, mengelompokkan objek di area tertentu dalam gambar, dan mengklasifikasikan objek berdasarkan fitur yang diekstraksi. Memahami metode deteksi objek yang lebih canggih, seperti SSD(Pan et al., 2020), memerlukan pemahaman konsep dasar ini.

- **Tinjauan singkat tentang algoritma Deep Learning yang digunakan dalam SSD.**

Dalam SSD, algoritma Deep Learning digunakan untuk mempelajari representasi-fitur yang mewakili objek dalam gambar dan membuat prediksi lokasi dan kelas objek dalam satu langkah. Itu juga menggunakan arsitektur jaringan saraf

tiruan yang dalam untuk mengekstraksi pola-pola yang kompleks dari data.

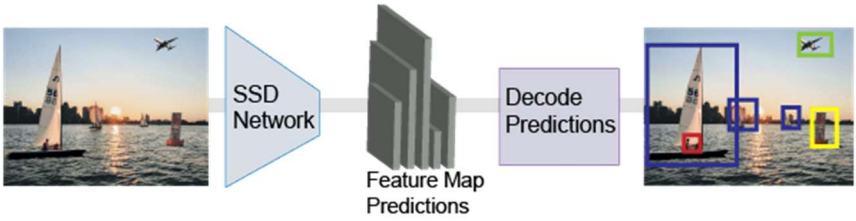
- **Pemahaman tentang jaringan konvolusi dan pentingnya dalam pengolahan citra.**

Salah satu jenis jaringan saraf tiruan yang sangat efektif dalam pengolahan gambar adalah jaringan saraf konvolusi, juga dikenal sebagai CNN(Indira et al., 2023; Jiao et al., 2020; Kattenborn et al., 2021). CNN menggunakan lapisan-lapisan konvolusi untuk mengekstraksi fitur-fitur yang relevan dari gambar. Pola-pola penting seperti tepi, tekstur, dan bentuk objek dapat diidentifikasi oleh CNN dengan melakukan konvolusi pada input gambar menggunakan filter-fitur yang terdefinisi. Karena keunggulannya, jaringan konvolusi menjadi pilihan utama dalam banyak aplikasi deteksi objek, termasuk SSD.

3. Pemahaman tentang SSD Multibox

- **Pengenalan tentang SSD (Single Shot Multibox Detector).**

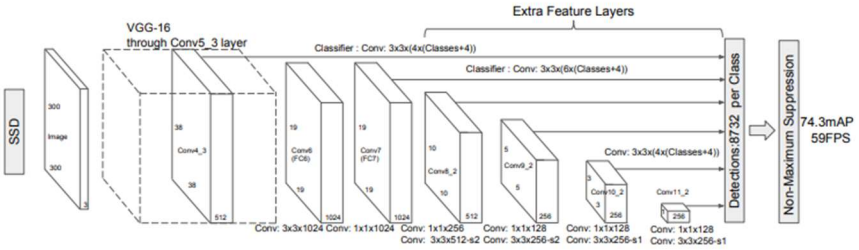
SSD Multibox, singkatan dari Detector Multibox Single Shot, adalah evolusi dari metode deteksi objek yang revolusioner. Berbeda dengan metode deteksi objek lainnya yang membutuhkan beberapa langkah untuk menemukan objek, SSD Multibox memiliki kemampuan untuk melakukan deteksi dalam satu langkah (single shot), yang membuatnya menjadi pilihan yang efektif dan efisien untuk aplikasi waktu nyata. Teknik ini memungkinkan deteksi objek secara bersamaan dalam berbagai skala dan aspek. Ini tanpa memerlukan proses yang kompleks seperti penggunaan skala multi-pyramid.



Gambar 15. Pengenalan gambar SSD Networks
Sumber : www.mathworks.com

- **Penjelasan tentang arsitektur SSD.**

Arsitektur SSD Multibox didasarkan pada konsep SSD yang terkenal, tetapi disempurnakan untuk menampilkan lebih banyak objek dalam gambar. Ini dicapai dengan menggunakan kotak pembatas, juga dikenal sebagai "prior boxes" atau "anchor boxes"(Luo et al., 2020), yang digunakan untuk memasukkan berbagai skala dan aspek objek yang digambarkan. Arsitektur ini terdiri dari lapisan konvolusi dengan tujuan untuk mengekstraksi fitur yang semakin abstrak dari gambar. Lokasi dan kelas objek kemudian diprediksi melalui lapisan ini.



Gambar 16. Pengenalan Struktur SSD / Sumber : www.researchgate.com

- Rincian tentang bagaimana SSD bekerja secara keseluruhan

Pendekatan multi-skala untuk deteksi objek digunakan oleh SSD Multibox. Pertama, jaringan konvolusi dimasukkan ke dalam gambar input, yang menghasilkan kumpulan fitur yang diwakili dalam berbagai tingkat resolusi. Untuk mendeteksi objek pada skala dan aspek yang berbeda, setiap fitur pada setiap tingkat resolusi dihubungkan ke serangkaian konvolusi yang berbeda. Output dari setiap lapisan konvolusi digunakan untuk membuat prediksi lokasi dan kelas objek.
Metode ini memungkinkan SSD Multibox(Cai et al., 2022; Lu et al., 2020; Yin et al., 2021) mendeteksi objek dalam gambar dengan berbagai skala dan aspek dan menghasilkan hasil deteksi yang cepat dan akurat. SSD Multibox sangat disukai karena kecepatan dan akurasi deteksinya, dan banyak digunakan dalam berbagai aplikasi deteksi objek saat ini.

4. Proses Implementasi

- **Pengenalan dataset yang digunakan untuk pelatihan.**

Sebelum memulai proses implementasi SSD Multibox, sangat penting untuk memilih dan menyiapkan dataset untuk pelatihan model. Dataset ini harus mencakup berbagai kelas objek yang ingin dideteksi serta berbagai variasi pose, skala, dan latar belakang. Contoh dataset yang biasa digunakan adalah dataset Pascal VOC, COCO, atau Open Images. Biasanya, dataset ini dilengkapi dengan anotasi yang mencakup lokasi dan kelas setiap objek dalam gambar.

- **Proses pra-pemrosesan data.**

Segera setelah dataset dipilih, proses pra-pemrosesan data dimulai. Ini mencakup pembacaan dataset, normalisasi gambar, pembagian dataset menjadi set pelatihan dan set validasi, dan pembuatan generator data untuk mengirimkan batch data ke model selama proses pelatihan. Untuk

menghindari overfitting, Anda juga dapat melakukan augmentasi data untuk meningkatkan variasi data pelatihan.

- **Pembuatan model SSD menggunakan Python dan kerangka kerja Deep Learning (misalnya, TensorFlow atau PyTorch).**

Setelah data diproses, langkah berikutnya adalah membuat model SSD menggunakan Python dan kerangka kerja Deep Learning seperti TensorFlow atau PyTorch. Model SSD terdiri dari beberapa lapisan konvolusi yang berfungsi untuk mengekstraksi fitur penting dari gambar, serta lapisan detektor yang berfungsi untuk memprediksi lokasi dan kelas objek dalam gambar. Anda dapat menggunakan arsitektur SSD yang tersedia dan menyesuaikannya dengan dataset Anda, atau Anda dapat merancang arsitektur yang berbeda sesuai dengan kebutuhan proyek Anda.

Pembuatan model SSD melibatkan sejumlah proses penting untuk membangun arsitektur deteksi objek yang efisien.

Berikut adalah rincian lebih lanjut tentang prosedur tersebut:

1. Arsitektur Model: Mendefinisikan arsitektur model SSD adalah langkah pertama. Anda perlu menentukan berapa banyak lapisan konvolusi, ukuran filter, dan jumlah filter, antara lain. Anda juga harus menentukan berapa banyak dan seberapa besar kotak pembatas yang akan digunakan untuk memprediksi lokasi objek.
2. Implementasi Lapisan Konvolusi: Selanjutnya, Anda harus menerapkan lapisan konvolusi yang digunakan untuk mengekstraksi fitur dari gambar. Ini biasanya mencakup penggunaan lapisan konvolusi, lapisan aktivasi (seperti ReLU), dan lapisan pooling untuk mengurangi dimensi fitur.
3. Pembuatan Lapisan Detektor: Setelah lapisan konvolusi, Anda perlu membuat lapisan detektor yang

akan memprediksi lokasi dan kelas objek dalam gambar. Ini biasanya memerlukan penggunaan beberapa lapisan konvolusi terakhir di jaringan untuk menghasilkan prediksi yang akurat.

4. Pengaturan Fungsi Kerugian: Setelah model dibangun, Anda harus menentukan fungsi kerugian, juga dikenal sebagai fungsi kerugian, yang akan digunakan selama pelatihan model. Komponen biasanya termasuk dalam fungsi kerugian ini untuk mengukur kesalahan prediksi lokasi objek dan prediksi kelas objek (seperti kesalahan cross-entropy).
5. Pelatihan Model: Anda dapat memulai pelatihan model menggunakan dataset yang telah diproses sebelumnya setelah model dibangun dan fungsi kerugian ditentukan. Proses pelatihan melibatkan mengoptimalkan parameter model untuk meminimalkan nilai fungsi kerugian dengan menggunakan algoritma optimisasi seperti Stochastic Gradient Descent (SGD) atau Adam.
6. Evaluasi dan Penyetelan: Setelah model dilatih, Anda perlu mengevaluasi kinerjanya menggunakan set data validasi yang berbeda. Ini melibatkan menghitung metrik evaluasi seperti akurasi, presisi, dan recall. Jika kinerja model kurang memuaskan, Anda dapat melakukan penyetelan hyperparameter atau peningkatan data untuk meningkatkan kinerjanya.
7. Inferensi: Setelah model dilatih dan dievaluasi, Anda dapat menggunakannya untuk melakukan inferensi pada gambar-gambar yang baru dibuat. Ini termasuk memasukkan gambar sebagai input ke model dan membuat perkiraan tentang lokasi dan kelas objek yang ada di gambar.

- **Contoh kode program dalam pembuatan model**

```
import tensorflow as tf
from tensorflow.keras import layers, models
input_shape = (224, 224, 3)  # Misalnya, gambar input ukuran 224x224 dengan 3 channel warna (RGB)
# Definisi arsitektur model
ssd_model = models.Sequential()
# Lapisan-lapisan konvolusi untuk ekstraksi fitur
ssd_model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape))
ssd_model.add(layers.MaxPooling2D((2, 2)))
ssd_model.add(layers.Conv2D(64, (3, 3), activation='relu'))
ssd_model.add(layers.MaxPooling2D((2, 2)))
ssd_model.add(layers.Conv2D(128, (3, 3), activation='relu'))
ssd_model.add(layers.MaxPooling2D((2, 2)))
# Lapisan-lapisan detektor untuk prediksi lokasi objek
ssd_model.add(layers.Flatten())
ssd_model.add(layers.Dense(256, activation='relu'))
ssd_model.add(layers.Dense(4))  # 4 parameter untuk lokasi objek (x_min, y_min, x_max, y_max)
# Contoh penggunaan
ssd_model.summary()
```

6. Pelatihan Model

- **Persiapan data untuk pelatihan.**

Perencanaan data pelatihan adalah langkah pertama dalam pelatihan model SSD. Data harus terdiri dari gambar yang diannotasi dengan kotak pembatas, atau kotak pembatas, yang menunjukkan lokasi objek dalam gambar dan label yang menunjukkan kelas objek yang terdapat dalam

setiap kotak pembatas. Untuk mengukur kinerja model selama pelatihan, bagikan dataset menjadi set pelatihan dan validasi.

- **Konfigurasi pelatihan model SSD.**

Mengonfigurasi pelatihan model SSD adalah langkah selanjutnya setelah data dipersiapkan. Ini termasuk memilih fungsi kerugian (loss function) yang sesuai, memilih algoritma optimisasi (seperti Adam atau Stochastic Gradient Descent), dan memilih metrik evaluasi yang akan digunakan selama pelatihan. Untuk mendapatkan hasil terbaik, Anda juga harus mengubah hyperparameter seperti kecepatan belajar, ukuran batch, dan jumlah epoch.

- **Proses pelatihan dan penyetelan hyperparameter.**

Anda dapat memulai proses pelatihan model setelah menyelesaikan konfigurasi pelatihan. Untuk melatih model menggunakan algoritma optimisasi yang telah dipilih, gunakan set pelatihan yang telah disiapkan sebelumnya. Selama pelatihan, mereka mengawasi metrik evaluasi seperti fungsi kehilangan dan akurasi pada set validasi untuk mengevaluasi kinerja model secara berkala. Untuk meningkatkan performa, sesuaikan hyperparameter, seperti mengubah rasio belajar atau mengubah arsitektur model.

7. Evaluasi dan Peningkatan Model

- **Metode evaluasi performa model.**

Beberapa metrik yang umum digunakan untuk mengevaluasi kinerja model deteksi objek seperti SSD termasuk:

1. Presisi Rata-Rata (AP): AP mengukur presisi rata-rata dari semua kelas objek yang terdeteksi. Ini adalah metrik yang umum digunakan untuk mengevaluasi performa model deteksi objek.

2. Presisi Rata-Rata (mAP): mAP adalah rata-rata AP dari semua kelas objek yang terdeteksi. Ini memberikan gambaran yang lebih luas tentang bagaimana model deteksi objek bekerja.
3. Intersection over Union (IoU): Ukuran seberapa baik kotak pembatas yang diprediksi dan yang sebenarnya tumpang tindih. Nilai IoU yang tinggi menunjukkan tingkat deteksi yang tinggi.

- **Strategi peningkatan kinerja model SSD.**

Beberapa pendekatan yang dapat dipertimbangkan untuk meningkatkan kinerja model SSD termasuk:

1. Augmentasi Data: Pengembangan data dapat membantu model mempelajari pola yang lebih umum dan meningkatkan variasi data pelatihan. Ini dapat mencakup menggerakkan gambar, memotongnya, bergerak, atau mengubah warnanya.
2. Penambahan Data: Model dapat belajar pola yang lebih kompleks dan umum dengan menambahkan lebih banyak data pelatihan. Anda dapat mencoba untuk menambah data dari sumber lain atau meningkatkan data yang sudah ada.
3. Peningkatan Arsitektur Model: Anda dapat mencoba memperluas atau memperdalam arsitektur model SSD untuk meningkatkan kapasitas model dan kemampuan untuk menangkap fitur yang lebih kompleks.

- **Pemecahan masalah umum selama evaluasi.**

Beberapa masalah umum yang mungkin timbul selama evaluasi model SSD meliputi:

1. Underfitting: Underfitting terjadi ketika model tidak cukup kompleks untuk mempelajari pola-pola dalam data pelatihan dan tidak dapat dengan baik menggeneralisasi data baru. Untuk mengatasi masalah

ini, Anda dapat menggunakan teknik regularisasi seperti pengurangan atau penyesuaian ulang hyperparameter.

2. Overfitting: Overfitting terjadi ketika model terlalu banyak "menghafal" data pelatihan dan tidak dapat menggeneralisasi data baru dengan baik. Untuk mengatasi masalah ini, Anda dapat menggunakan teknik regularisasi seperti dropout atau penyesuaian ulang hyperparameter.
3. Kesalahan Anotasi: Anotasi data pelatihan yang salah dapat menyebabkan hasil evaluasi yang salah. Sebelum memulai proses evaluasi, pastikan untuk memeriksa dan memvalidasi secara menyeluruh anotasi data pelatihan.

8. Penerapan dan Kasus Penggunaan

- Pada penerapan ini terdapat beberapa file yang dapat diunduh pada url github berikut ini : https://github.com/proxip/ssd-tester
- Contoh kode program :

```
1. import cv2
2. import matplotlib.pyplot as plt
3. config_file = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
4. frozen_model = 'frozen_inference_graph.pb'
5. model = cv2.dnn_DetectionModel(frozen_model, config_file)
6. classLabels = []
7. filename = 'labels.txt'
8. with open(filename, 'rt') as spt:
   classLabels = spt.read().rstrip('\n').split('\n')
   model.setInputSize(320, 320)
```

```
    model.setInputScale(1.0/127.5)
    model.setInputMean((127.5, 127.5, 127.5))
    model.setInputSwapRB(True)
9.  img = cv2.imread('file.png')
10. classIndex, confidence, bbox = model.detect(img,
    confThreshold=0.5)
11. font = cv2.FONT_HERSHEY_PLAIN
12. for classInd, conf, boxes in zip(classIndex.flatten(),
    confidence.flatten(), bbox):
    cv2.rectangle(img, boxes, (255, 0, 0), 2)
    cv2.putText(img, classLabels[classInd-1], (boxes[0] +
    10, boxes[1] + 40), font, fontScale = 3, color=(0, 255,
    0), thickness=3)
13. plt.figure(figsize=(10, 10))
14. plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
15. plt.axis('off')
16. plt.show()
17. cv2.imwrite('result.png', img)
```

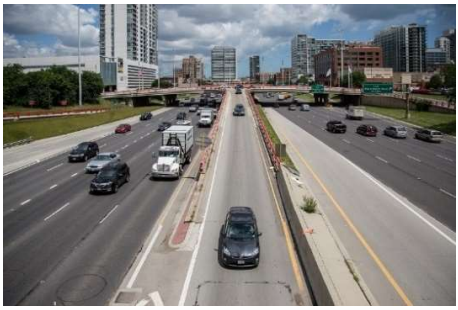
- **Keterangan pada kode program yang perlu diperhatikan**

1. config_file = harus disesuaikan dengan file berekstensi .pbtxt
2. frozen_model = harus disesuaikan dengan file berekstensi .pb
3. filename = harus disesuaikan dengan file bernama labels.txt
4. img = harus disesuiakan dengan file yang akan diuji dengan menggunakan ekstensi .png
5. Setiap kode program yang tidak memiliki nomor adalah kode program dengan indent atau menjorok kedalam.

Namun untuk setiap kode yang memiliki nomor adalah kode program baru pada setiap baris.

- **Gambar asli**

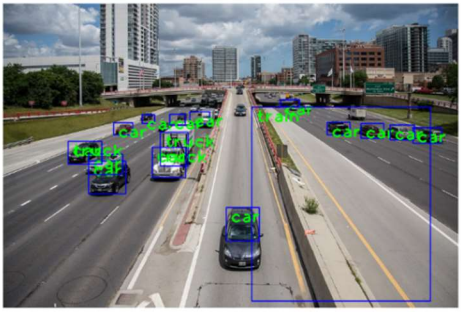
Berikut adalah gambar asli yang digunakan pada saat melakukan uji coba, dimana gambar masih asli tanpa adanya kotak klasifikasi.



Gambar 17. Tampilan gambar sebelum klasifikasi sumber : www.wikipedia.org

- **Gambar hasil prediksi**

Setelah kode program diatas dijalankan, maka akan muncul kotak – kotak yang menampikan beberapa jenis klasifikasi, seperti gambar mobil yang ditampilkan.



Gambar 18. Tampilan gambar setelah klasifikasi, sumber : dokumen penulis

DAFTAR PUSTAKA

Anushka, Arya, C., Tripathi, A., Singh, P., Diwakar, M., Sharma, K., Pandey, H., 2021. Object Detection using Deep Learning: A Review, in: Journal of Physics: Conference Series. IOP Publishing Ltd.

Cai, J., Makita, Y., Zheng, Y., Takahashi, S., Hao, W., Nakatoh, Y., 2022. Computers and Electrical Engineering 104.

Indira, D.N.V.S.L.S., Goddu, J., Indraja, B., Challa, V.M.L., Manasa, B., 2023. Mater Today Proc 80, 3438–3443.

Jiao, J., Zhao, M., Lin, J., Liang, K., 2020. Neurocomputing 417, 36–63.

Kattenborn, T., Leitloff, J., Schiefer, F., Hinz, S., 2021. Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. ISPRS Journal of Photogrammetry and Remote Sensing.

Kaur, R., Singh, S., 2022. A comprehensive review of object detection with deep learning. Digital Signal Processing: A Review Journal.

Lu, H., Li, C., Chen, W., Jiang, Z., 2020. Pattern Recognit Lett 140, 295–302.

Luo, Q., Ma, H., Tang, L., Wang, Y., Xiong, R., 2020. Neurocomputing 378, 364–374.

Pan, H., Jiang, J., Chen, G., 2020. Signal Process Image Commun 89.

Yin, Q., Yang, W., Ran, M., Wang, S., 2021. Signal Process Image Commun 98.

BAB 8
COMPUTER VISION DENGAN YOLO

Oleh Aji Seto Arifianto, S.ST, M.T

1. Pendahuluan YOLO

Deteksi objek merupakan salah satu topik computer vision yang banyak menarik minat peneliti selama beberapa dekade terakhir. Hal ini berkaitan dengan identifikasi dan lokalisasi objek dalam gambar atau video, yang umumnya digunakan di berbagai aplikasi praktis, seperti pengawasan keamanan, kendaraan otomatis, pengenalan wajah, pengolahan citra medis, dan banyak lagi. Deteksi objek menghadapi sejumlah tantangan, termasuk variasi pose, skala, rotasi, pencahayaan, dan latar belakang yang kompleks. Sebagai contoh, dalam sebuah gambar jalan raya, deteksi objek harus mampu mengenali dan melokalisasi berbagai jenis kendaraan, pejalan kaki, tanda lalu lintas, dan objek lainnya dalam berbagai kondisi pencahayaan dan cuaca.

Untuk mengatasi tantangan ini, banyak peneliti telah mengembangkan berbagai metode dan teknik dalam deteksi objek, mulai dari pendekatan berbasis fitur seperti Histogram of Oriented Gradients (HOG) dan Haar-like features hingga pendekatan berbasis deep learning seperti Convolutional Neural Networks (CNN), selain itu akhir-akhir ini metode YOLO cukup popular bahkan berkembang dengan berbagai versi algoritmanya.

2. Metode YOLO

YOLO merupakan pendekatan untuk deteksi objek memakai jaringan syaraf tunggal, yang melakukan prediksi bounding box serta probabilitas kelas secara langsung dalam satu penilaian(Putra et al., 2023). YOLO membagi foto jadi grid dengan dimensi SxS. Tiap sel grid melakukan prediksi terhadap Bounding Boxes B serta nilai kepercayaan (Box confidence scores) untuk kotak tersebut dan probabilitas kelas C. Prediksi Bounding Boxes B terdiri dari 5 komponen, yaitu x, y, w, h, serta Box confidence score.

- Koordinat (x,y) adalah representasi pusat kotak relatif terhadap batas- batas kotak grid. Koordinat ini dinormalisasi supaya titik tersebut terletak di antara 0 serta 1.
- Ukuran kotak (w,h) merupakan dimensi gambar (yang dinormalisasi).
- Nilai Box confidence score mencerminkan tingkatan keyakinan jika Bounding Boxes B berisi sesuatu objek, dan tingkatan akurasi prediksi tersebut. Oleh sebab itu, prediksi YOLO mempunyai wujud vektor output[S, S, B×5+C].

Perhitungan box confidence score atau nilai confidence untuk sebuah kotak ditunjukkan pada persamaan berikut (Redmon et al., 2016)

$$Box\ confidence\ score\ = P_r(object) \cdot IoU_{pred}^{truth}$$

$$IoU_{pred}^{truth} = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

dimana

$\boldsymbol{P_r(object)}$ **:** probabilitas kotak berisi objek, jika ada objek bernilai 1, jika tidak bernilai 0.

$\boldsymbol{IoU_{pred}^{truth}}$ **:** intersection over union antara kotak prediksi dan kebenaran dasar (ground truth)

Class confidence score memberi nilai kepercayaan kelas secara spesifik untuk setiap kotak, yang memberikan kemungkinan kelas yang muncul di kotak dan seberapa sesuainya kotak yang diprediksi dengan objek. Persamaan pada class confidence score sebagai berikut

$$P_r\left(Class_i|object\right) \cdot box\ confidence\ score = P_r\left(Class_i\right) \cdot IoU_{pred}^{truth}$$

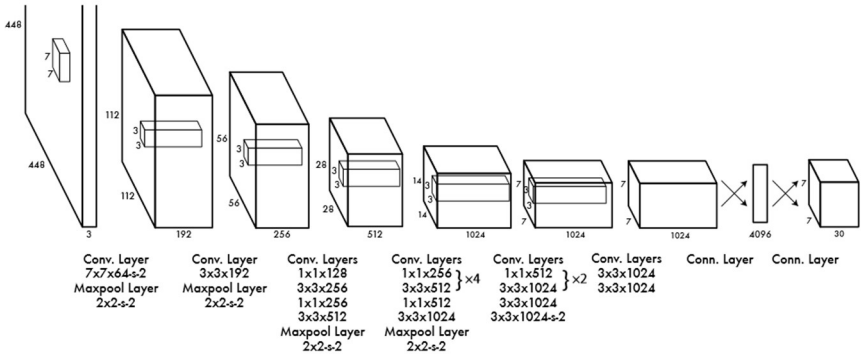
dimana

$\boldsymbol{P_r\left(Class_i|object\right)}$: probabilitas kondisional kelas i.

$\boldsymbol{P_r\left(Class_i\right)}$: probabilitas kelas i.

3. Arsitektur Jaringan YOLO (You Only Look Once)

Arsitektur YOLO dikembangkan untuk mengoptimalkan kecepatan, akurasi, dan efisiensi. Berdasarkan publikasi Joseph Redmon dkk (2016) dasar arsitektur YOLO menggunakan 24 layer konvolusi dengan 2 layer koneksi(Redmon et al., 2016).



Gambar 19. Arsitektur Dasar YOLO

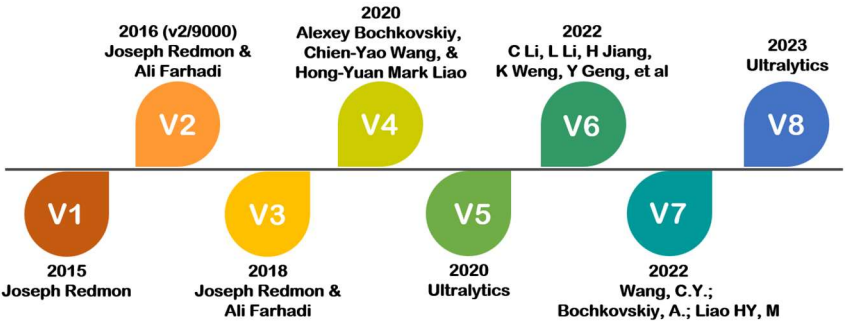
Arsitektur YOLO memiliki beberapa fitur utama:

1. **Single Neural Network**: Arsitektur YOLO terdiri dari satu rangkaian neural network yang terintegrasi sepenuhnya. Jaringan ini menerima gambar input dan menghasilkan prediksi kotak pembatas (bounding boxes) serta probabilitas kelas objek dalam satu langkah feedforward. Pendekatan ini membedakan YOLO dari beberapa metode deteksi objek lain yang memerlukan beberapa langkah untuk melakukan prediksi.
2. **Convolutional Layers**: YOLO menggunakan lapisan konvolusi untuk mengekstraksi fitur dari gambar input. Lapisan konvolusi ini membantu jaringan untuk memahami konteks visual dari gambar dan menemukan pola yang relevan untuk deteksi objek.
3. **Grid Cell**: Gambar input dibagi menjadi sebuah grid cell yang membentang di seluruh gambar. Setiap sel dalam grid ini bertanggung jawab untuk memprediksi objek yang mungkin ada di dalamnya. Melalui proses ini, YOLO dapat secara efisien menemukan objek di berbagai lokasi dalam gambar.

4. **Predicted Bounding Boxes**: Setiap sel dalam grid memprediksi beberapa kotak pembatas (bounding boxes) yang mencakup objek. Untuk setiap kotak pembatas, YOLO menghasilkan koordinat relatif terhadap ukuran sel tersebut serta probabilitas kelas objek yang diprediksi.
5. **Non-Max Suppression**: Setelah menghasilkan prediksi kotak pembatas, YOLO menerapkan teknik non-maximum suppression (NMS) untuk mengurangi redundansi dalam prediksi. NMS memastikan bahwa hanya kotak pembatas yang paling relevan dan memiliki tingkat kepercayaan tertinggi yang disimpan sebagai hasil deteksi akhir.
6. **Loss Function**: Arsitektur YOLO menggunakan loss function yang dirancang khusus, seperti sum of squared error (SSE) atau cross-entropy loss, untuk mengukur kesalahan prediksi jaringan. Loss function ini membimbing jaringan untuk memperbarui bobotnya selama proses pelatihan dengan tujuan meningkatkan akurasi deteksi objek.

4. Perkembangan YOLO v1 hingga v8

Model-model YOLO terkenal karena kecepatan dan efisiensinya dalam mendeteksi objek dalam satu kali pemrosesan (single shot). Mari kita bahas perkembangan YOLO dari versi 1 hingga versi 8(Hussain, 2023; Terven et al., 2023)



Gambar 20. YOLO Timeline

1. YOLOv1

Arsitektur YOLOv1 menggunakan arsitektur jaringan konvolusi tunggal yang memproses gambar dalam satu kali pass. Salah satu model tercepat pada masanya, mampu memproses **45 frame per detik**. Kelemahannya akurasi lebih rendah pada objek kecil dan kesulitan dalam mendeteksi objek yang tumpang tindih.

2. YOLOv2 (YOLO9000)

YOLOv2 memperkenalkan batch normalization, anchor boxes, dan ukuran input yang dapat disesuaikan untuk meningkatkan akurasi. YOLO9000 merupakan versi yang dapat mendeteksi lebih dari 9000 kelas dengan menggabungkan deteksi objek dan klasifikasi hierarkis. Kelebihannya meningkatkan kecepatan tanpa mengorbankan akurasi yang signifikan.

3. YOLOv3

YOLOv3 memperkenalkan FPN (Feature Pyramid Network) untuk mendeteksi objek pada berbagai skala dengan lebih baik. Menggunakan arsitektur backbone Darknet-53 yang lebih dalam dan lebih efisien. Menunjukkan peningkatan kinerja baik dalam hal kecepatan dan akurasi dibandingkan dengan YOLOv2.

4. YOLOv4

Menggunakan berbagai teknik (seperti CSPDarknet, Mish activation, SPP, PANet, dan lainnya) untuk meningkatkan kinerja tanpa mengurangi kecepatan. YOLOv4 menghasilkan peningkatan signifikan dalam akurasi dan efisiensi.

5. YOLOv5

Dibangun dengan PyTorch yang mempermudah pelatihan dan pengembangan lebih lanjut. Menyediakan berbagai ukuran model (s,m,l,x) untuk menyeimbangkan antara kecepatan dan akurasi. Community-driven, banyak diadopsi oleh komunitas karena kemudahan penggunaannya dan dokumentasi yang baik.

6. YOLOv6

Fokus pada peningkatan kinerja untuk Optimalisasi industri. YOLOv6 lebih efisien dalam pemanfaatan sumber daya komputasi dibandingkan dengan YOLOv5.

7. YOLOv7

YOLOv7 melakukan revolusi arsitektur, memperkenalkan teknik baru dalam jaringan CNN untuk deteksi objek. Menggabungkan keunggulan YOLOv4 dan YOLOv5 untuk memberikan kinerja yang lebih baik.

8. YOLOv8

Arsitektur YOLOv8 menggunakan backbone CSPDarknet53 yang telah dimodifikasi. Modul C2f menggantikan CSPLayer yang digunakan dalam YOLOv5. Lapisan spatial pyramid pooling fast (SPPF) mempercepat komputasi dengan memadukan fitur ke dalam peta berukuran tetap. Setiap konvolusi dilengkapi dengan normalisasi batch dan aktivasi SiLU.

5. Proses Pelatihan YOLO

1. **Persiapan Dataset**: Langkah pertama dalam pelatihan YOLO adalah persiapan data latih. Data latih harus berisi gambar-gambar yang telah dianotasi dengan kotak pembatas (bounding boxes) yang menunjukkan lokasi dan label dari objek yang ingin dideteksi. Anotasi ini diperlukan untuk membimbing jaringan dalam pembelajaran (supervised learning).
2. **Inisialisasi Bobot**: Sebelum pelatihan dimulai, bobot (weights) dalam jaringan YOLO perlu diinisialisasi. Inisialisasi ini bisa dilakukan secara acak atau menggunakan bobot dari jaringan yang telah dilatih sebelumnya (pre-trained weights).
3. **Optimisasi Bobot**: Selama pelatihan, jaringan YOLO memperbarui bobotnya dengan menggunakan algoritma optimisasi seperti stochastic gradient descent (SGD) atau varian-varian lainnya seperti Adam atau RMSprop. Tujuan optimisasi ini adalah untuk meminimalisir loss function antara prediksi jaringan dan label yang sebenarnya dari data latih.
4. **Propagasi Maju (Forward Propagation)**: Pada setiap iterasi pelatihan, gambar-gambar dari data latih dimasukkan ke dalam jaringan YOLO. Jaringan melakukan forward propagation untuk menghasilkan prediksi kotak pembatas (bounding box predictions) dan probabilitas kelas objek untuk setiap gambar.
5. **Perhitungan Loss**: Setelah prediksi dibuat, loss function dihitung untuk mengukur seberapa besar kesalahan antara prediksi jaringan dan anotasi yang sebenarnya. Loss function ini mencakup komponen-komponen seperti

kesalahan lokalisasi (seperti IoU loss) dan kesalahan klasifikasi (seperti cross-entropy loss).

6. **Propagasi Mundur (BackPropagation)**: Setelah perhitungan loss, langkah berikutnya adalah backpropagation. Dalam langkah ini, gradien loss dihitung kembali melalui jaringan menggunakan algoritma backpropagation. Gradien ini digunakan untuk memperbarui bobot dalam jaringan, dengan tujuan mengurangi loss pada iterasi berikutnya.
7. **Penyesuaian Hyperparameter**: Proses pelatihan YOLO juga melibatkan penyesuaian hyperparameter seperti learning rate, batch size, dan jumlah iterasi. Hyperparameter ini mempengaruhi kecepatan dan kualitas konvergensi jaringan selama pelatihan.
8. **Validasi dan Evaluasi**: Setelah pelatihan selesai, jaringan YOLO dievaluasi menggunakan data uji yang terpisah untuk mengukur kinerjanya. Matrik evaluasi yang umum digunakan adalah mean Average Precision (mAP) dan IoU (Intersection over Union) untuk mengevaluasi akurasi deteksi objek.

6. Studi Kasus

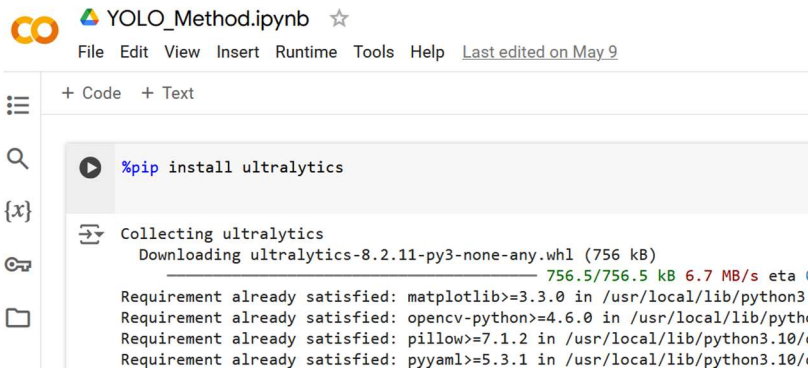
Studi kasus kali ini, akan diberikan contoh mendeteksi objek kendaraan truk menggunakan YOLOv8.

1. Siapkan data citra/gambar/video yang akan digunakan untuk dataset aplikasi computer vision. Kali ini penulis mencontohkan kasus deteksi kendaraan truk dengan sumber data dari rekaman CCTV di Jalan Raden Intan Kota Malang (http://cctv.malangkota.go.id/cameras_dua) data yang awalnya berbentuk video kemudian di-sampling menjadi citra/gambar (bisa menggunakan aplikasi converter video to image). Dataset kemudian dibagi

menjadi data latih (train) dan data uji (test). Untuk data latih harus melalui proses Anotasi terlebih dulu (Bab 5 Persiapan dan Prapemrosesan Data).

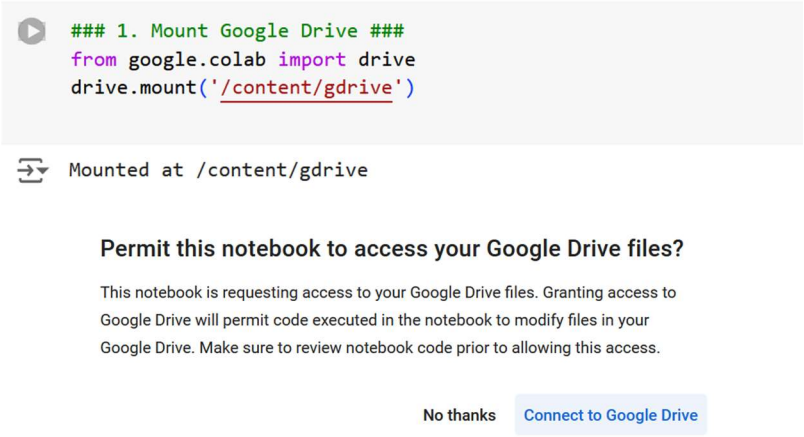
Gambar 21. Contoh Data Train

2. Untuk mengembangkan aplikasi computer vision, penulis menggunakan google colab sebagai platform berbasis cloud untuk menulis dan menjalankan bahasa pemrograman python. Pembaca bisa menggunakan tool lain seperti kaggle atau editor berbasis desktop seperti PyCharm, Jupyter Notebook dan lainnya.
3. Karena menggunakan YOLO v8, langkah pertama harus meng-install lebih dulu library dari ultralytics, gunakan perintah %pip install ultralytics. Tunggu beberapa saat sampai proses selesai.



Gambar 22. Contoh Perintah install library dari ultralitics

4. Hubungkan (mounted) google colab dengan google drive Anda.



Gambar 23. Contoh Perintah mounted dan permit

Setelah berhasil, maka di jendela kiri akan terlihat ada folder baru bernama gdrive

5. Arahkan root_directory ke lokasi tempat menyimpan dataset, dengan perintah seperti berikut ini

```
root_directory = '/content/gdrive/MyDrive/Colab Notebooks/Truck Detection'
```

Contoh path milik penulis: /content/gdrive/MyDrive/Colab Notebooks/Truck Detection

6. Langkah berikutnya melakukan pelatihan/train Model dengan YOLOv8 tipe nano. Pembaca bisa memilih tipe lain sesuai kebutuhan.

```
### Train model ###
import os
from ultralytics import YOLO

# Load a model
model = YOLO("yolov8n.pt")  # load pre trained model v8m, v8n, v8s, v8l, v8x
# Use the model
results = model.train(data=os.path.join(root_directory, "data.yaml"), epochs=10)
```

- **Yolov8n.pt** → tipe nano
  (tipe dan spesifikasi yolov8 dapat dilihat di https://docs.ultralytics.com/models/yolov8/#performance-metrics)
- Didalam contoh ini menggunakan 10 epochs, Anda bisa merubah nilai epoch untuk hasil yang lebih baik.
- **data.yaml** → file hasil export dataset yang sudah dianotasi dari roboflow

Jalankan, tunggu hingga proses selesai.

```
                from  n    params  module                                       arguments
  0                -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
  5                -1  1     73984  ultralytics.nn.modules.conv.Conv             [64, 128, 3, 2]
  6                -1  2    197632  ultralytics.nn.modules.block.C2f             [128, 128, 2, True]
  7                -1  1    295424  ultralytics.nn.modules.conv.Conv             [128, 256, 3, 2]
  8                -1  1    460288  ultralytics.nn.modules.block.C2f             [256, 256, 1, True]
  9                -1  1    164608  ultralytics.nn.modules.block.SPPF            [256, 256, 5]
 10                -1  1         0  torch.nn.modules.upsampling.Upsample         [None, 2, 'nearest']
```

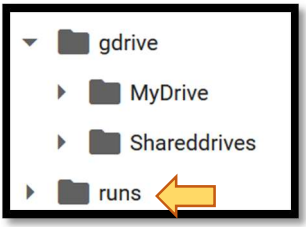
Berikut contoh 1 epoch sekitar 3 menit 34

```
Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
 1/10         0G      1.284      2.682        1.5         14        640: 100%|██████████| 14/14 [03:34<00:00, 15.31s/it]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 1/1 [00:08<00:00,  8.80s/it]
```

Berikut contoh tampilah jika proses pelatihan model Yolo telah selesai (dengan durasi 0,622 menit untuk 10 epochs)

```
10 epochs completed in 0.622 hours.
Optimizer stripped from runs/detect/train/weights/last.pt, 6.2MB
Optimizer stripped from runs/detect/train/weights/best.pt, 6.2MB

Validating runs/detect/train/weights/best.pt...
Ultralytics YOLOv8.2.11 🚀 Python-3.10.12 torch-2.2.1+cu121 CPU (Intel Xeon 2.20GHz)
Model summary (fused): 168 layers, 3005843 parameters, 0 gradients, 8.1 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 1/1 [00:05<00:00,  5.89s/it]
                   all         21         28      0.925      0.964       0.96       0.77
Speed: 2.7ms preprocess, 247.2ms inference, 0.0ms loss, 8.9ms postprocess per image
Results saved to runs/detect/train
```

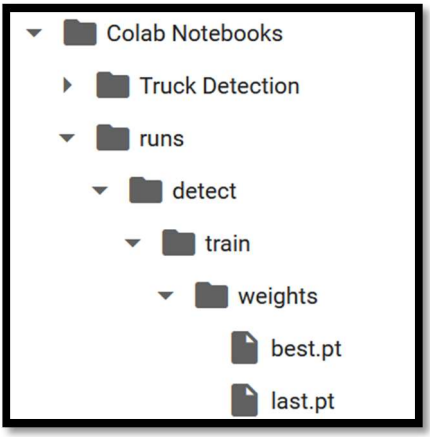
Serta ditandai dengan munculnya folder **runs** diluar **gdrive**



7. Selanjutnya, menyalin hasil training ke googledrive (perhatikan alamat path lokasi file salinan)

```
### Copy results ###
!scp -r /content/runs '/content/gdrive/MyDrive/Colab Notebooks'
```

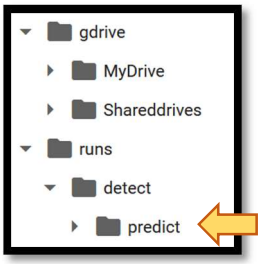
Kita dapat melihat hasil training/pelatihan model di folder: Colab Notebooks/runs/detect/train/weights



8. Setelah train model selesai, kita lakukan prediksi terhadap satu contoh citra masukkan. Untuk menguji model hasil pelatihan perlu memperhatikan path modelnya (lokasi dari model YOLO, default namanya **best.pt**)

```
[10] ultralytics import YOLO
     = YOLO('/content/gdrive/MyDrive/Colab Notebooks/runs/detect/train/weights/best.pt')
    .predict('/content/gdrive/MyDrive/Colab Notebooks/Truck Detection/test/images/Test1.jpg', save = True)
```

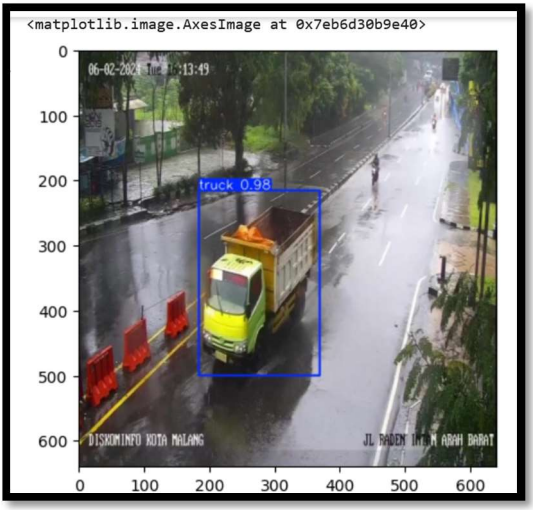
Hasil prediksi masuk ke folder **runs/detect/predict** diluar folder **gdrive**



9. Menampilkan hasil prediksi dengan menambahkan library **matplotlib** lebih dulu kemudian menuliskan perintah berikut

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
# Load an image from a file
img = mpimg.imread('/content/runs/detect/predict/Test1.jpg')
# Display the image
plt.imshow(img)
```

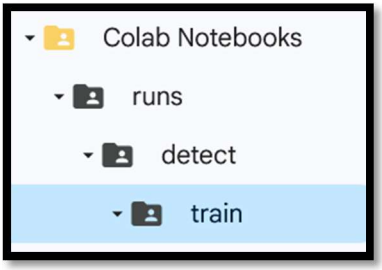
(path /content/runs/detect/predict/NamaCitraUji.jpg)



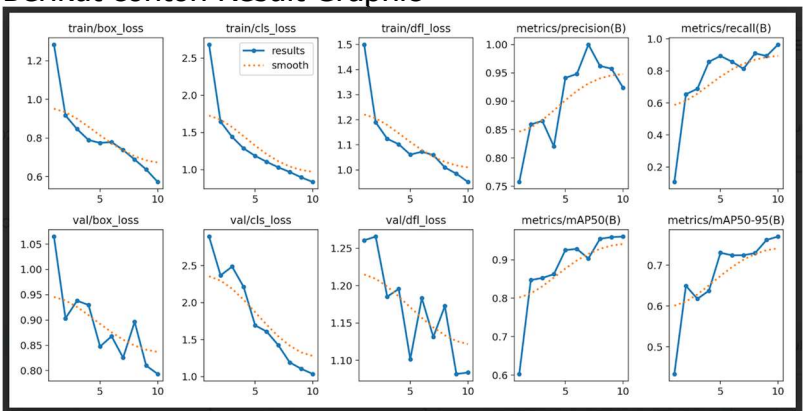
Gambar 24. Hasil prediksi Model Yolov8 berhasil mendeteksi

Berikut hasil prediksi kendaraan truk ditandai dengan bounding boxes dan nilai probabilitasnya.

10. Untuk melihat hasil pelatihan/train model YOLOv8 yang dilakukan sebelumnya dapat membuka folder **train** seperti berikut



Berikut contoh Result Graphic



Gambar 25. Result Graphic

7. Implementasi YOLO di Berbagai Bidang

- **Keamanan dan Pengawasan**: YOLO telah digunakan dalam sistem keamanan dan pengawasan untuk mendeteksi objek mencurigakan, orang, kendaraan, dan aktivitas yang tidak biasa(Wang et al., 2021).
- **Mobil Otonom**: Dalam konteks mobil otonom, YOLO digunakan untuk mendeteksi kendaraan, pejalan kaki, tanda lalu lintas, dan objek lainnya di sekitar mobil. Ini membantu mobil otonom dalam navigasi dan pengambilan keputusan yang aman(Redmon & Farhadi, 2018).

- **Kesehatan dan Kedokteran**: YOLO digunakan dalam bidang kesehatan dan kedokteran untuk mendeteksi dan memvisualisasikan struktur anatomi dalam gambar medis, seperti deteksi organ dalam citra CT scan(Liu et al., 2020)
- **Industri Manufaktur**: YOLO digunakan dalam industri manufaktur untuk mendeteksi cacat pada produk, memantau jalannya proses produksi, dan menjaga kualitas produk secara keseluruhan. Seperti pemantauan kualitas kemasan produk (Vu et al., 2022).
- **Pertanian**: Dalam bidang pertanian, YOLO dapat digunakan untuk mendeteksi dan mengklasifikasikan berbagai objek, seperti tanaman, hama, atau penyakit pada tanaman(Mohanty et al., 2016).

8. Kesimpulan

Berikut beberapa keunggulan YOLO:

1. YOLO dapat mendeteksi secara real-time dan responsif untuk gambar maupun video dengan berbagai berkualitas.
2. Arsitektur YOLO dirancang menghasilkan model yang lebih ringan, deteksi objek lebih akurat dan konsisten serta mudah diimplementasikan.
3. Model YOLO dapat diadaptasi dengan baik untuk deteksi objek diberbagai konteks dan kondisi yang kompleks.
4. Model YOLO tersedia secara open-source, memungkinkan siapa saja untuk berkontribusi dalam pengembangannya.

Namun YOLO juga memiliki keterbatasan, antara lain:
1. YOLO sering kali lemah untuk mendeteksi objek yang saling tumpang tindih. YOLO juga cenderung sulit untuk mendeteksi bbjek yang kecil.

2. Kinerja YOLO sangat bergantung pada kualitas dan representasi data pelatihan yang digunakan. Jika dataset pelatihan kurang lengkap dalam hal variasi, maka kinerja YOLO juga kurang baik.
3. Meskipun arsitekturnya efisien, YOLO tetap mengonsumsi jumlah memori yang besar, terutama pada model dengan lapisan-lapisan neural network yang banyak. Ini dapat menjadi masalah pada perangkat dengan sumber daya terbatas, seperti perangkat mobile atau sistem embedded.

Catatan Tambahan

Selain menggunakan Google Colab, Anda dapat menggunakan Kaggle untuk membuat pelatihan model YOLO secara online, proses train di kaggle memang lebih cepat daripada google collab (perbandingan dengan dataset dan dan epoch sama) kaggle membutuhkan waktu 1,02 menit sedangkan google collab 37,32 menit. Di Kaggle menyediakan GPU tidak berbayar sehingga proses cepat, sedangkan Google collab GPU berbayar. Untuk dataset kaggle dapat mengambil dari roboflow atau unggah manual, seadangkan di google collab dataset unggah manual.

DAFTAR PUSTAKA

Hussain, M. (2023). YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection. In *Machines* (Vol. 11, Issue 7). https://doi.org/10.3390/machines11070677

Liu, C., Hu, S. C., Wang, C., Lafata, K., & Yin, F. F. (2020). Automatic detection of pulmonary nodules on CT images with YOLOv3: development and evaluation using simulated and patient data. *Quantitative Imaging in Medicine and Surgery*, *10*(10). https://doi.org/10.21037/QIMS-19-883

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, *7*(September). https://doi.org/10.3389/fpls.2016.01419

Putra, P. Y., Arifianto, A. S., Fitri, Z. E., & Puspitasari, T. D. (2023). Deteksi Kendaraan Truk pada Video Menggunakan Metode Tiny-YOLO v4. *Jurnal Informatika Polinema*, *9*(2). https://doi.org/10.33795/jip.v9i2.1243

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-December*. https://doi.org/10.1109/CVPR.2016.91

Redmon, J., & Farhadi, A. (2018). YOLO v.3. *Tech Report*, 1–6. https://pjreddie.com/media/files/papers/YOLOv3.pdf%0Ahttps://pjreddie.com/yolo/.

Terven, J., Córdova-Esparza, D. M., & Romero-González, J. A.

(2023). A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. In *Machine Learning and Knowledge Extraction* (Vol. 5, Issue 4). https://doi.org/10.3390/make5040083

Vu, T. T. H., Pham, D. L., & Chang, T. W. (2022). A YOLO-based Real-time Packaging Defect Detection System. *Procedia Computer Science*, *217*. https://doi.org/10.1016/j.procs.2022.12.285

Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2021). Scaled-yolov4: Scaling cross stage partial network. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR46437.2021.01283

BAB 9
COMPUTER VISION DENGAN REGRESSION

OLeh Giandari Maulani, S.Kom, M.Kom

1. Pendahuluan

Computer Vision merupakan Perpaduan dari Pengolahan Gambar/Citra (*Image processing*) dan Pengenalan Pola (*Pattern Recognition*), sedangkan Regression atau Regresi Linier merupakan salah satu analisis regresi dalam metode numerik dan merupakan analisis statistika yang memodelkan beberapa variabel dalam hubungan persamaan linier. Jika keduanya digabung, maka akan memperoleh hasil yang tepat dan akurat, berupa data maupun informasi yang dibutuhkan dan dapat membantu pekerjaan manusia yang biasanya dilakukan secara manual atau tanpa teknologi. Penerapan Computer Vision dengan Regression dapat diterapkan dalam berbagai bidang kehidupan manusia, yang dapat memberikan banyak manfaat, serta prediksi-prediksi yang tepat.

2. Konsep Teoritis Computer Vision dan Regression

a. Computer Vision

Computer Vision merupakan suatu metode penggunaan komputer dengan tujuan untuk mendapatkan informasi penting dari sebuah Citra/Gambar Digital. (A.A.M.Suradi, dkk. 2022)

Computer Vision merupakan kemampuan komputer untuk menampilkan objek digital, koleksi datanya secara visual dan dapat membantu pekerjaan-pekerjaan yang sulit dilakukan oleh manusia. (Wibowo, A.P.W. 2016)

Computer Vision merupakan bidang ilmu komputer yang berfokus pada kemampuannya untuk 'Melihat', yang serupa dengan Penglihatannya manusia.
Karena penglihatan berkaitan erat dengan Pencahayaan, maka pencahayaan inipun dapat mempengaruhi kualitas hasil 'Penglihatan Komputer'. (Wahyudi, D.A. 2011)

Computer Vision merupakan Perpaduan dari Pengolahan Gambar/Citra (*Image processing*) dan Pengenalan Pola (*Pattern Recognition*). Tujuan utama Computer Vision yakni untuk pembuatan model, *extract data*, serta perolehan informasi yang dibutuhkan yang berasal dari Gambar. (Munawar, Z. 2023).

Computer Vision memiliki kemampuan menganalisis gambar dari objek, yang kemudian Hasilnya dapat memberikan manfaat bagi kita (manusia), kemampuan Computer Vision sebagai berikut :

1) Computer Vision memiliki kemampuan menganalisis Piksel dan mengenali Pola.
2) Computer Vision memiliki kemampuan *Class Prediction* dari Gambar.
3) Computer Vision memiliki kemampuan *Extraction of Information* dari Gambar. (Purno, A & Wibowo, W. 2016)

Aplikasi Computer Vision saat ini banyak diterapkan pada berbagai bidang, salah satu contohnya seperti pada bidang-bidang dibawah ini :

1) Pada bidang Robotika ; Computer Vision yang diterapkan pada bidang ini untuk mengontrol robot dalam meniru gerakan dan deteksi kecepatan robot dalam bergerak.
2) Pada bidang Pendidikan ; Computer Vision yang diterapkan pada bidang ini untuk membantu para pengajar dalam hal pelacakan kehadiran siswa, untuk pemetaan ruang kelas dan ruang lainnya secara Real-time, untuk keamanan sekolah, untuk pencegahan Vandalisme, dll.
3) Pada bidang Pertanian ; Computer Vision yang diterapkan pada bidang ini untuk mengontrol kualitas tanaman, mendeteksi kerusakan tanaman, penyemprotan pestisida, serta untuk penyortiran tanaman.
4) Pada bidang Pertahanan dan Keamanan ; Computer Vision yang diterapkan pada bidang ini untuk mendeteksi keberadaan ranjau, mendeteksi wajah, serta untuk mengontrol kendaraan militer yang tidak berawak.
5) Pada bidang Otomotif ; Computer Vision yang diterapkan pada bidang ini untuk mengontrol kendaraan otonom agar dapat bernavigasi melintasi jalan dengan benar dan untuk mengembangkan sistem autopilot.

6) Pada bidang Retail/Bisnis eceran ; Computer Vision yang diterapkan pada bidang ini untuk mengontrol jumlah stok barang di rak, kualitas dan kuantitas barang yang dijual, melacak data *customer behavior*, serta untuk manajemen inventaris barang/produk.
7) Pada bidang Perawatan Kesehatan ; Computer Vision yang diterapkan pada bidang ini untuk mengidentifikasi berbagai macam penyakit dan diagnosa yang tepat, serta untuk

pengambilan gambar/citra medis dengan tingkat keakuratan yang tinggi.

8) Pada bidang Mode/*Fashion* ; Computer Vision yang diterapkan pada bidang ini untuk menganalisa pakaian yang tepat dan sesuai untuk konsumen dengan berbasis gambar, untuk perkiraan *Fashion Trends* yang akan terjadi, serta rekomendasi mode/fashion yang tepat bagi konsumen.

Pada bidang Asuransi ; Computer Vision yang diterapkan pada bidang ini untuk menganalisa aset, untuk menghitung jumlah kerugian, untuk penanganan *Claim* pelanggan, dll.

3. Regression atau Regresi Linier

Regression atau Regresi Linier merupakan salah satu Analisis regresi dalam metode numerik dan merupakan analisis statistika yang memodelkan beberapa variabel dalam hubungan persamaan linier. Regresi Linier mempunyai bentuk persamaan yang tampak seperti di bawah ini :

(1) $$y' = a + bx_i$$

(2) $$b = \frac{n\Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n\Sigma x_i^2 - (\Sigma x_i)^2}$$

(3) $$a = \bar{y_l} - b\bar{x_l}$$

Keterangan:
y' = variabel dependen sistem (cm)
a = Konstanta
b = Koefisien variabel x
x_i = variabel independen (piksel)
y_i = variabel dependen sebenarnya (cm)
$\bar{y_l}$ = Rata-rata variabel $\bar{y_l}$
$\bar{x_l}$ = Rata-rata variabel $\bar{x_l}$

Gambar 26. Bentuk Persamaan Regresi Linier
Sumber : Abadi, A.B.,dkk. (2022)

Regresi Linier memiliki 2 (dua) *Type*, yakni Regresi Linier Berganda dan yang satunya lagi yakni Regresi Linier Sederhana.

Regresi Linier Berganda memiliki beberapa (lebih dari 1) variabel bebas dan variabel terikat, sedangkan Regresi Linier Sederhana memiliki 1 variabel bebas dan 1 variabel terikat.

Analisis Regresi Linier merupakan metode Statistik yang umumnya dipakai dalam berbagai penelitian dan untuk penelitian berbasis komputer, statistiknya menggunakan SPSS.

Fungsi analisis Regresi Linier antara lain :

a. Untuk Efisiensi ; Regresi Linier memiliki fungsi untuk meningkatkan efisiensi bagi perusahaan/instansi di dalam menjalankan operasional perusahaan /instansinya tersebut, dengan penyajian hasil perhitungan regresi linier yang akurat.
b. Untuk *Future Prediction* ; Regresi Linier memiliki fungsi untuk memprediksi masa depan yang terkait dengan Peluang dan Resiko. Peluang disini maksudnya peluang bisnis yang tepat, dan resiko maksudnya Regresi Linier juga dapat melihat berbagai resiko yang dapat terjadi dari keputusan yang sudah diambil.
c. Untuk Perbaikan Kesalahan ; Jika keputusan yang diambil oleh perusahaan ternyata salah, maka Regresi Linier akan memperbaikinya dengan Perhitungan-perhitungan sehingga hasilnya dapat diperoleh.
d. Untuk memberikan Pengetahuan Baru ; Regresi Linier memiliki fungsi untuk memberikan pengetahuan terbaru bagi perusahaan/instansi.

Contoh dari analisis Regresi Linier yakni untuk memproyeksikan pengembalian saham serta menghasilkan

biaya modal. Pengembalian saham diregresikan untuk pengembalian indeks yang lebih luas untuk menghasilkan Beta untuk saham-saham tertentu. Beta merupakan Resiko saham yang terkait dengan Indeks Pasar.

4. Penelitian yang menerapkan Computer Vision dengan Regression

Penerapan Computer Vision dengan Regression dapat diterapkan dalam berbagai bidang yang dapat memberikan banyak manfaat. Dibawah ini beberapa Penelitian yang menerapkan Computer Vision dan Regression/Regresi Linier :

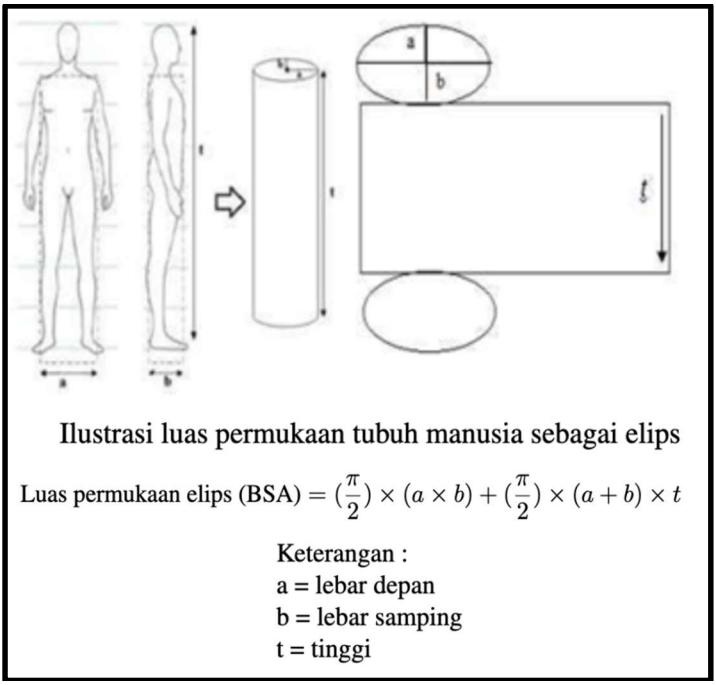
a. Perhitungan Indeks Massa Tubuh (IMT) dengan mengurangi kontak langsung (*Less Contact*) berbasis Computer Vision dan Regresi Linear.

Pada penelitian ini, penerapan Computer Vision dan Regression/Regresi Linier digunakan untuk perhitungan Indeks Massa Tubuh (IMT) dengan mengurangi kontak langsung (*Less Contact*) dengan banyak orang. Indeks Massa Tubuh (IMT) ini didapatkan dengan membandingkan berat dan tinggi badan seseorang. Penelitian ini merancang sebuah sistem perhitungan Indeks Massa Tubuh (IMT) dengan mengkombinasikan Computer Vision dan Regression/Regresi Linier sehingga menghasilkan perhitungan yang otomatis dengan menggunakan Sensor Kamera yang dapat mengurangi kontak langsung (*Less Contact*) secara efektif dan efisien, dengan cara sebagai berikut: Untuk tahap pertama yakni pengambilan gambar tampak depan dan tampak samping tubuh manusia menggunakan Kamera. Tahap Kedua yakni pengolahan gambar dengan *Grayscale,* Deteksi Tepi, *Blur,* serta *Bounding Box* untuk menghasilkan nilai berat, lebar dan

tinggi badan dalam Piksel. Tahap Ketiga yakni melakukan operasi Regresi Linier yang dapat mengkonversi nilai piksel ke dalam Centimeter untuk Tinggi dan Lebar badan. Untuk mendapatkan hasil perhitungan Berat badan, penelitian ini menggunakan metode BSA (*Body Surface Area*).

BSA (*Body Surface Area*) adalah Kalkulasi luas area tubuh manusia menggunakan pendekatan rumus tabung pada tubuh manusia. (Abadi, A.B.,dkk. 2022).

Rumus tersebut digambarkan seperti ini :



Ilustrasi luas permukaan tubuh manusia sebagai elips

$$\text{Luas permukaan elips (BSA)} = (\frac{\pi}{2}) \times (a \times b) + (\frac{\pi}{2}) \times (a + b) \times t$$

Keterangan :
a = lebar depan
b = lebar samping
t = tinggi

Gambar 27. Rumus perhitungan Body Surface Area (BSA) Sumber : Abadi, A.B.,dkk. (2022)

BSA (*Body Surface Area*) juga merupakan cara yang dilakukan untuk memperoleh data dan informasi mengenai kondisi sebenarnya dari Tubuh seseorang. (Alfian, M.I.,dkk. 2019). Rumus lainnya dari BSA (*Body Surface Area*) yakni dengan rumus *Mosteller* yang dipublikasikan di tahun 1987.

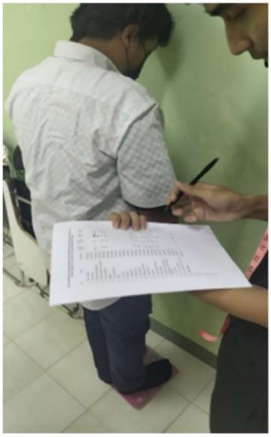
$$BSA(m^2) = \sqrt{\frac{Height(cm) \times Weight(kg)}{3600}}$$

Gambar 28. Rumus Body Surface Area (BSA) Mosteller
Sumber : Karim, M.I.M., dkk. (2020)

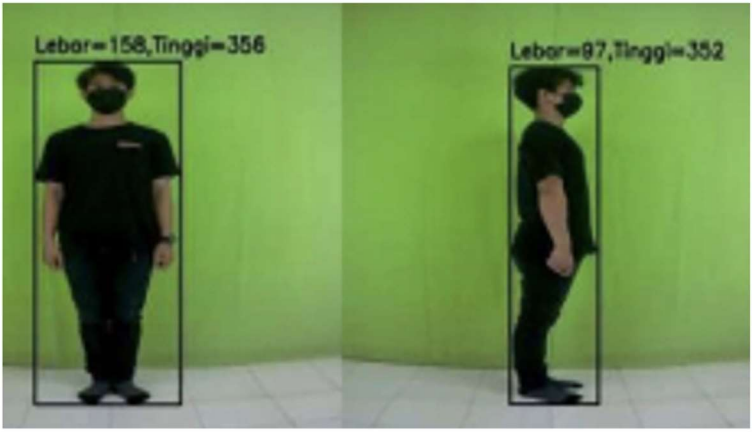
Penelitian yang menerapkan Computer Vision dan Regression/Regresi Linier untuk perhitungan Indeks Massa Tubuh (IMT) dengan mengurangi kontak langsung (*Less Contact*) ini menghasilkan :

a. Perolehan tingkat keakuratan perhitungan Berat Badan sebesar 88,54% dengan rata-rata kesalahan/*Error* sebesar 11,4%.
b. Perolehan tingkat keakuratan perhitungan Tinggi Badan sebesar 98,96% dengan rata-rata kesalahan/*Error* sebesar 1,04%.
c. Perolehan Skor Indeks Massa Tubuh (IMT) sebesar 88,24% dengan rata-rata kesalahan/*Error* sebesar 11,74%.

Perolehan Nilai Akurasi untuk kategori Indeks Massa Tubuh (IMT) yang dihasilkan Sistem dengan Indeks Massa Tubuh (IMT) Kondisi Sebenarnya, didapat sebesar 60%.



Pengukuran Berat Badan dengan menggunakan Timbangan.

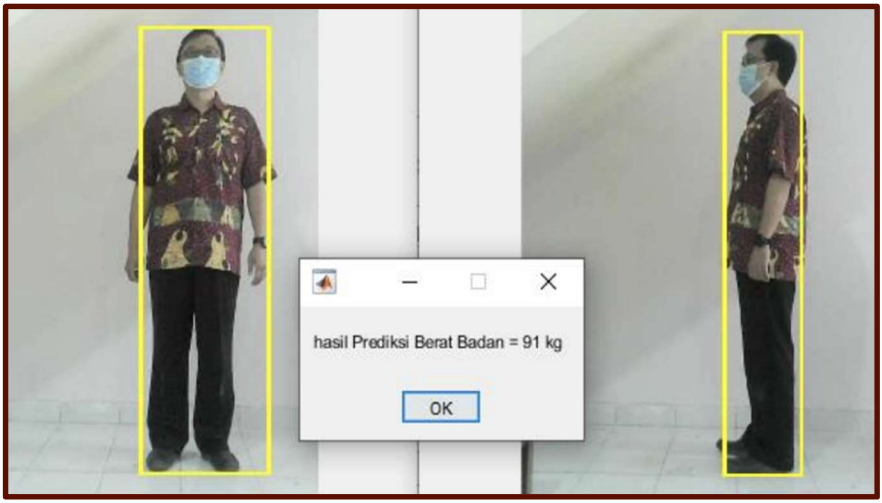


Pengukuran Berat Badan dengan Computer Vision dan Regression/Regresi Linier.

Gambar 29. Pengukuran berat badan dengan Timbangan dan pengukuran dengan menggunakan Kamera, Computer Vision dan Regression/Regresi Linier. Sumber : Abadi, A.B.,dkk. (2022)

b. Prototype alat pengukur berat dan tinggi badan seseorang dengan berbasis Computer Vision menggunakan Regresi Linier.

Pada penelitian ini, penerapan Computer Vision dan Regression/Regresi Linier digunakan untuk mengetahui dan melakukan perhitungan berat dan tinggi badan seseorang Tanpa harus menggunakan alat timbang badan.
Berat dan tinggi badan seseorang merupakan Besaran Fisik yang hasilnya didapatkan dengan mengukur seseorang tersebut. Hasil pengukurannya digunakan untuk berbagai macam keperluan yang berhubungan dengan data berat badan seseorang dalam Kilogram dan data tinggi badan seseorang dalam Centimeter. Selama ini pengukuran berat dan tinggi badan seseorang masih dilakukan secara manual dengan tenaga manusia. Masalah timbul saat terjadinya Pandemi Covid-19 dimana kontak fisik antara seseorang dengan orang lainnya harus diminimalkan untuk mengurangi penularan dan penyebaran penyakit Covid-19 tersebut. Untuk mengurangi masalah tersebut, maka dibuatlah penelitian ini, yang menerapkan Computer Vision dan Regression/Regresi Linier untuk mengetahui dan melakukan perhitungan berat dan tinggi badan seseorang Tanpa harus menggunakan alat timbang badan, namun dengan menggunakan Foto. Foto ini kemudian dikombinasikan dengan Computer Vision dan Regression/regresi Linier, yang tidak memerlukan kontak fisik antar orang, yang dapat bekerja dengan otomatis dan memiliki hasil yang akurat. Penelitian ini kemudian membuat sebuah Prototype alat pengukur berat dan tinggi badan seseorang berdasarkan Foto tersebut.



Gambar 30. Hasil Prediksi Berat Badan dilihat dari tampak depan dan tampak samping dengan Regresi Linier. Sumber : Firmansyah, R. (2020)

Penelitian yang menerapkan Computer Vision dan Regression/Regresi Linier untuk mengetahui dan melakukan perhitungan berat dan tinggi badan seseorang tanpa harus menggunakan alat timbang badan ini menghasilkan :
Jumlah rata-rata kesalahan/*Error* yang ditemukan dari penggunaan Prototype alat ukur berat badan dengan menerapkan Computer Vision dan Regresi Linier yang dilakukan dengan MAPE sebesar 13,8 %. (Firmansyah, R. 2020)

MAPE/Mean Absolute Percent Error adalah Rata-rata Persentase Kesalahan secara absolut/mutlak yang memiliki akurasi prediksi tinggi dengan menggunakan pengukuran statistik. MAPE menyajikan informasi tentang jumlah/besarnya kesalahan pengukuran (dalam Persentase) dari Nilai *Real*-nya. Jika nilai kesalahannya semakin kecil, maka hasil prediksinya semakin akurat. (Khoiri, 2020)

c. Pengukuran panjang dan berat pada Ikan (Lele) dan Sayuran (Kangkung) pada Budikdamber menggunakan Computer Vision dan regresi Linier.

Pada penelitian ini, penerapan Computer Vision dan Regression digunakan untuk melakukan prediksi berapa ukuran panjang ikan (lele) dan sayuran (kangkung) beserta ukuran beratnya. Hal ini penting didalam Budikdamber/Budi daya ikan dalam ember. Untuk melakukan prediksi ini digunakan perangkat Kamera. Metode Computer Vision diterapkan dalam mengekstraksi fitur jumlah piksel suatu objek untuk posisi tampak atas, sedangkan Regression/Regresi Linier diterapkan dalam prediksi panjang serta berat objek, yang dalam hal ini yakni ikan (lele) dan sayuran (kangkung).

Penelitian yang menerapkan Computer Vision dan Regression pada Budikdamber ini menghasilkan :

1. Prediksi pada panjang dan berat Ikan (Lele) memiliki hasil tepat dan akurat.
2. Prediksi pada panjang dan berat Sayuran (Kangkung) masih perlu memiliki tingkat keakuratan yang rendah/kurang akurat, sehingga perlu adanya perbaikan untuk penambahan atau perubahan fitur.

Banyaknya jumlah Ikan (Lele) dan Sayuran (Kangkung) akan mempengaruhi hasil, sehingga jika jumlah Ikan (Lele) dan Sayuran (Kangkung) yang diteliti ini Banyak, maka akan didapatkan simulasi prediksi yang hasilnya lebih baik dan akurat, sesuai dengan kondisi *Real* Budikdambernya. (Fitriyah, H. (2020).

Gambar 31. Budikdamber/Budidaya Ikan dalam Ember
Sumber : Fitriyah, H. (2020)

DAFTAR PUSTAKA

Abadi, A. B., Fadllullah, A., Sumardi, S., Mahdi, S., & Juniar, A. N. (2022). Perhitungan Indeks Massa Tubuh Less Contact Berbasis Computer Vision dan Regresi Linear. *MATRIK: Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *21*(3), 629-638.

Alfian, M.I., Fitriyah,H., Utaminingrum,F. (2019). Sistem pengukuran tinggi dan berat badan berdasarkan perhitungan Body Surface Area (BSA) menggunakan Bounding Box berbasis Raspberry Pi. *Jurnal Pengembangan Teknologi Informasi dan ilmu Komputer, 3*(6), 5242-5249.

Fitriyah, H. (2020). Pengukuran Panjang-Berat Ikan dan Sayuran pada Budikdamber (Budi Daya Ikan dalam Ember) Menggunakan Visi Komputer dan Regresi Linier.

Jurnal SISKOM-KB (Sistem Komputer dan Kecerdasan Buatan), *4*(1), 8-14.

Khoiri. (2020). Cara menghitung Mean Absolute Percentage Error (MAPE) di Excel dan pengertiannya. https://www.khoiri.com/2020/12/pengertian-dan-cara-menghitung-mean-absolute-percentage-error-mape.html

Karim, M. I. M., Putra, H. F. T. S., & Magdalena, R. (2020). Analisis Pengolahan Citra Telapak Kaki Manusia Terhadap Berat Dan Tinggi Badan. *e-Proceedings of Engineering*, *7*(1).

Munawar, Z., Hasnawi, M., Beno, I. S., Darma, I. W. A. S., Sastradipraja, C. K., Sutramiani, N. P. & Koibur, M. E. (2023). *Visi Komputer: Konsep, Metode, dan Aplikasi*. Kaizen Media Publishing.

Purno, A., & Wibowo, W. (2016). Implementasi teknik computer vision dengan metode Colored Markers Trajectory secara real time. *Jurnal Teknik Informatika, 8*(1), 45-48.

Suradi, A. A. M., Rasyid, M. F., & Nasaruddin, N. (2022, August). Sistem Perhitungan Jumlah Kendaraan Berbasis Computer Vision. In *SISITI: Seminar Ilmiah Sistem Informasi dan Teknologi Informasi* (Vol. 11, No. 1, pp. 89-97).

Wibowo, A. P. W. (2016). Implementasi Teknik Computer Vision Dengan Metode Colored Markers Trajectory Secara Real Time. *Jurnal Teknik Informatika*, *8*(1), 38-42.

Wahyudi, D. A., & Kartowisastro, I. H. (2011). Menghitung Kecepatan Menggunakan Computer Vision. *Jurnal Computer Engineering Binus University*, *19*, 101.

BAB 10
COMPUTER VISION DENGAN CLUSTERING

Oleh Johar Nur Iin, S. Kom., MT

1. Pendahuluan

Clustering merupakan pengelompokan objek-objek ke dalam beberapa kategori yang berbeda. Hirarki dan non hirarki merupakan jenis metode clustering. Clustering merupakan metode yang kerap digunakan pada berbagai cabang ilmu seperti mechine learning, data science, statistika, pattern recognition, sorftware engineering dan artificial inteligence. Computer vision merupakan cabang dari kecerdasan buatan (artificial intelligence) yang berkaitan dengan pemrosesan dan analisis citra. Python merupakan bahasa pemrograman yang kerap digunakan pada masalah terkait computer vision karena menyediakan berbagai library guna memudahkan dalam analisis dan visualisasi data(Aladžuz et al., 2022). Pada bab ini akan dibahas pemanfaatan algoritma *clustering* pada *computer vision* dengan bahasa pemrograman python.

2. Clustering pada Computer Vision

Computer vision bekerja dengan mengadopsi penglihatan manusia pada satu atau lebih gambar guna merekonstruksi karakteristiknya, seperti bentuk, pencahayaan, dan distribusi warna (Carsten Steger, Markus Ulrich, 2018). *Computer Vision*, menggali informasi dari gambar (Dhabliya et al., 2023), kemudian berkembang

menafsirkan dan mengekstrak informasi dari audio dan video. Di sisi lain, *machine learning* memproses data yang telah ada guna menghasilkan fitur prediktif (Khan & Al-Habsi, 2020). Tujuan utama *computer vision* yaitu memanfaatkan gambar untuk membangun model dan menambang data dan informasi (Dhabliya et al., 2023). Implementasi clustering pada computer vision dapat dilakukan dengan melalui :

a. *Image segmentation* yaitu mengelompokkan citra digital ke dalam area terpisah sehingga piksel pada setiap area memilki kecocokan yang tinggi dan kontras yang tinggi antar area (Dhanachandra et al., 2015).
b. Color quantization, pada citra digital dapat dilakukan peminimalan jumlah warna, jumlah bit yang berkurang berdampak pada kompresi atau dapat meminimalkan ruang yang dibutuhkan untuk menyimpan citra digital. Hal ini karena setiap bit menjadi representasi warna pada setiap piksel (Tommy et al., 2021).
c. Object tracking , mencocokkan piksel untuk menentukan piksel yang memiliki kemiripan dengan objek yang terdeteksi pada camera, kemudian dilakukan pengelompokan semua anggota piksel ke dalam sebuah cluster berdasarkan fitur warna atau teksture untuk digunakan melokalisasi dan memperkirakan lokasi objek (Pradhana, 2014).
d. Image retrieval, clustering dapat membantu dalam *content base image retrieval* dengan memetakan gambar pada basisdata berdasarkan karakeristik fitur visualnya ke dalam berbagai cluster. Kemudian dilakukan pencocokan query gambar yang diberikan pengguna dengan cluster(Pradhana, 2014).

3. Deteksi objek menggunakan K-Means

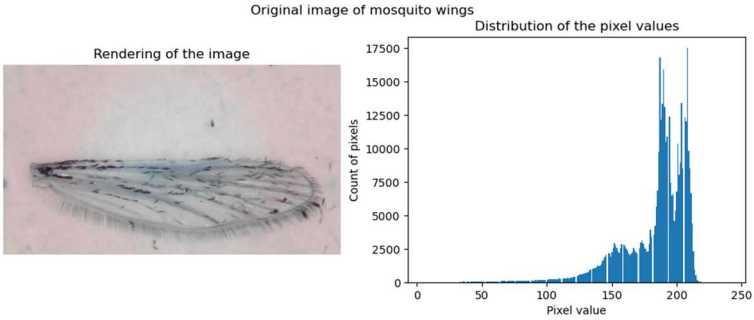
Salah satu aplikasi computer vision yang paling populer adalah deteksi objek dan klasifikasi objek. Pada bab ini akan dibahas deteksi objek pada python.

Tahap 1 : Mengimport Library yang akan digunakan

```
import pandas as pd
import numpy as np
from skimage import io
import matplotlib.pyplot as plt
import cv2
from sklearn.cluster import KMeans
from mpl_toolkits.mplot3d import Axes3D
import plotly as py
import plotly.graph_objs as go
import warnings
warnings.filterwarnings("ignore")
```

Tahap 2 : Membaca dan menampilkan file gambar

```
import matplotlib.pyplot as plt
image=cv2.imread('albojt1.png')
fig, ax = plt.subplots(ncols=2, figsize=(12, 4))

ax[0].imshow(image, cmap=plt.cm.gray)
ax[0].axis("off")
ax[0].set_title("Rendering of the image")
ax[1].hist(image.ravel(), bins=256)
ax[1].set_xlabel("Pixel value")
ax[1].set_ylabel("Count of pixels")
ax[1].set_title("Distribution of the pixel values")
_ = fig.suptitle("Original image of mosquito wings")
```



Gambar 32. Image dan Histogram dari Image

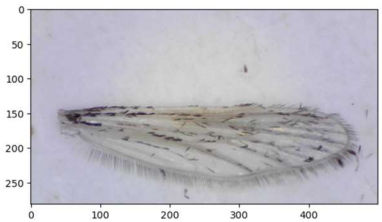
Tahap 3 : Properti gambar mencakup jumlah baris, kolom, dan chanel warna dari image; image dtype merupakan jenis data gambar yaitu uint8. img.shape akan mengembalikan tupel jumlah baris, kolom yaitu 281,500 dan chanel warna 3 merupakan represntasi dari *red*, *green* dan *blue*.

```
print(image.dtype,image.shape)
```

```
uint8 (281, 500, 3)
```

Tahap 4 : Konversi Gambar dari BGR ke RGB

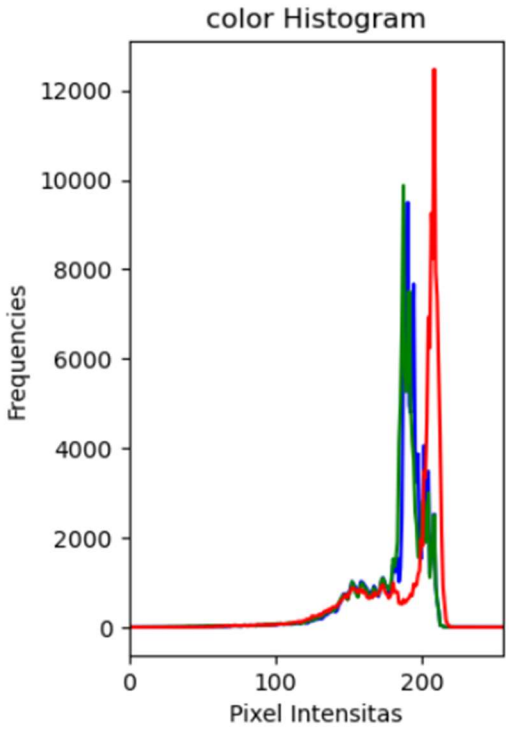
```
# Changing the image from BGR to RGB
image=cv2.cvtColor(image,cv2.COLOR_BGR2RGB)
plt.imshow(image);
#plt.axis('off') ;
```



Gambar 33. Hasil Image RGB

Tahap 5. Menampilkan color histogram dengan mendeteksi warna RGB dari image

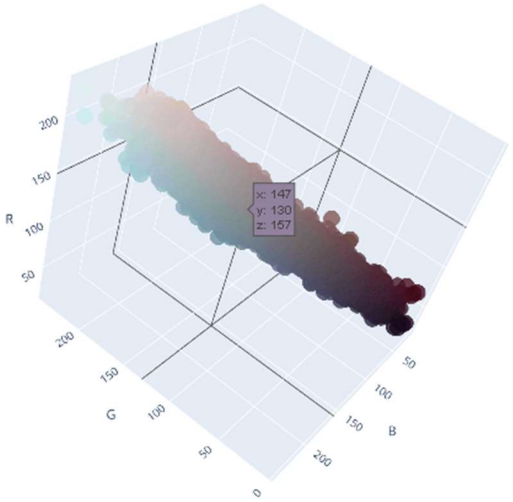
```
plt.subplot(1,2,2)
bluehist=cv2.calcHist([image],[0],None,[256],[0,255])
greenhist=cv2.calcHist([image],[1],None,[256],[0,255])
redhist=cv2.calcHist([image],[2],None,[256],[0,255])
plt.xlabel('Pixel Intensitas')
plt.xlim(0,255)
plt.ylabel("Frequencies")
plt.plot(bluehist,color="b")
plt.plot(greenhist,color="g")
plt.plot(redhist,color="r")
plt.title("color Histogram")
plt.show()
```



Gambar 34. Histogram Saluran RGB dari Image

Tahap 6. Mengubah chanel warna image menjadi tiga array numpy berbeda yang mewakili nilai piksel masing-masing saluran warna red (r), green (g) dan blue (b). flatten mengubah saluran warna yang berbentuk array 2 Dimensi menjadi larik 1 Dimensi. Kemudian menggunakan library plotly untuk menampilkan visualisasi sebaran nilai r,g, b.

```
# Assuming 'image' is defined
r, g, b = cv2.split(image)
r = r.flatten()
g = g.flatten()
b = b.flatten()
# Create a list of colors based on RGB values
colors = np.stack((r, g, b), axis=-1) / 255.0  # Normalize to [0, 1] range
trace1 = go.Scatter3d(
    x= r,
    y= g,
    z= b,
    mode='markers',
     marker=dict(
        color =colors ,
        size=10,
        line=dict(
            color= colors ,
            width= 10
        ),
        opacity=0.8
     )
)
data = [trace1]
layout = go.Layout(
       width=1000,   # Lebar plot dalam piksel
    height=1000,  # Tinggi plot dalam piksel
    title= 'Clusters',
    scene = dict(

            xaxis = dict(title  = 'R'),
            yaxis = dict(title  = 'G'),
            zaxis = dict(title  = 'B')
        )
)
fig = go.Figure(data=data, layout=layout)
py.offline.iplot(fig)
```



Gambar 35. Visualisasi Sebaran Saluran Warna RGB dari Image menggunakan Scatter Plot 3D

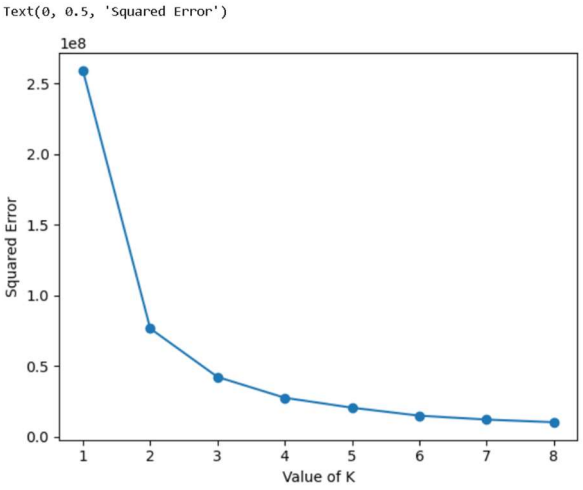
Tahap 7 : Mengubah array image menjadi bentuk array 2 Dimensi di mana jumlah baris yang terbentuk sebanyak jumlah piksel pada image. Kemudian mengubah array 2D yang bertipe data uin8 ke tipe data float32

```
# Reshaping the image to 2d
pixel_values = image.reshape((image.shape[0]*image.shape[1],3))
pixel_values = np.float32(pixel_values)
pixel_values
```

```
array([[192., 193., 191.],
       [193., 191., 191.],
       [198., 192., 193.],
       ...,
       [206., 192., 194.],
       [208., 191., 194.],
       [206., 192., 194.]], dtype=float32)
```

Tahap 8. Penentuan nilai optimal K dengan Metode *Elbow*. Metode ini bekerja dengan menguji stabilitas jumlah *cluster* terbaik dengan membandingkan perbedaan *Sum of Square Error* (SSE) masing-masing cluster, cluster terbaik diperoleh dari perbedaan paling signifikan yang membentuk siku (Umargono et al., 2020).

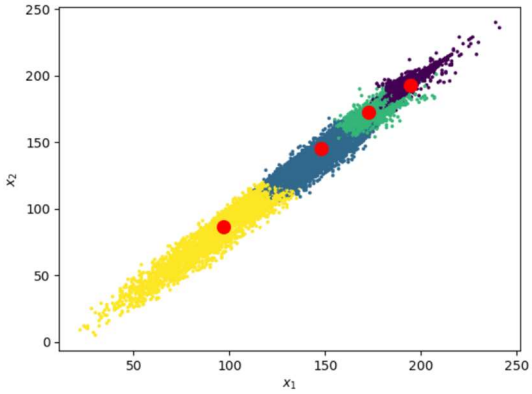
```
J=[]
for k in range (1,9):
    kmeans=KMeans(n_clusters=k)
    kmeans.fit(pixel_values)
    centroids=kmeans.cluster_centers_
    clusters=kmeans.predict(pixel_values)
    j=0
    for i in range(len(pixel_values)):
        center=centroids[clusters[i]]
        j+=(np.linalg.norm(pixel_values[i]-center))**2
    J.append(j)
plt.plot(range(1,9),J,'-o')
plt.xlabel("Value of K")
plt.ylabel("Squared Error")
```



Gambar 36. Metode Elbow untuk Nilai K Optimal

Tahap 9. Visualisasi cluster yang terbentuk di mana k=4

```
kmeans=KMeans(n_clusters=4)
clusters=kmeans.fit_predict(pixel_values)
centroids=kmeans.cluster_centers_
plt.scatter(pixel_values[:,0],pixel_values[:,1],s=3,c=clusters)
plt.scatter(centroids[:,0],centroids[:,1],s=100,c='r')
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.show()
```



Gambar 37. Scatter Plot Untuk Visualisasi hasil K Means

Tahap 10 Segmentasi image dengan K Means. Beberapa argument dibutuhkan untuk melakukan clustering dengan K Means.

cv.kmeans(data, K, bestLabels, criteria, attempts, flags[, centers])

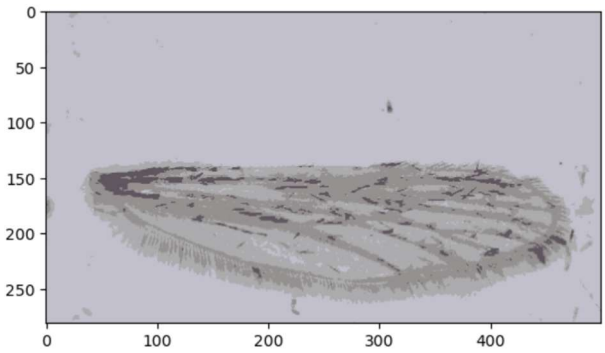
Data yaitu Sampel beripe data np.float32
K yaitu jumlah klaster yang akan dibentuk
Criteria yaitu menentukan berapa lama proses segmentasi bekerja ketika mencari letak cluster optimal.
Kriteria berhenti dengan 3 tipe berikut:

a. **cv.TERM_CRITERIA_EPS** yaitu perulangan algoritma berhenti jika akurasi yang ditentukan dan epsilon, tercapai.

b. **cv.TERM_CRITERIA_MAX_ITER** yaitu algoritma berhenti setelah jumlah perulangan maksimal ditentukan.
c. **cv.TERM_CRITERIA_EPS + cv.TERM_CRITERIA_MAX_ITER** yaitu ketika salah satu kondisi terpenuhi algoritma perulangan berhenti.

Flags : untuk menentukan bagaimana pusat awal cluster digunakan. Flag terdiri dari 2(dua) jenis yaitu cv.KMEANS_PP_CENTERS (inisialisasi pusat awal klaster dengan k Means ++) dan cv.KMEANS_RANDOM_CENTERS (memilih pusat awal klaster secara acak)

```
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 100, 0.2)

k = 4
retval, labels, centers = cv2.kmeans(pixel_values, k, None, criteria, 20, cv2.KMEANS_RANDOM_CENTERS)
centers = np.uint8(centers)
segmented_data = centers[labels.flatten()]

segmented_image = segmented_data.reshape((image.shape))
labels_reshape = labels.reshape(image.shape[0], image.shape[1])

plt.imshow(segmented_image);
```

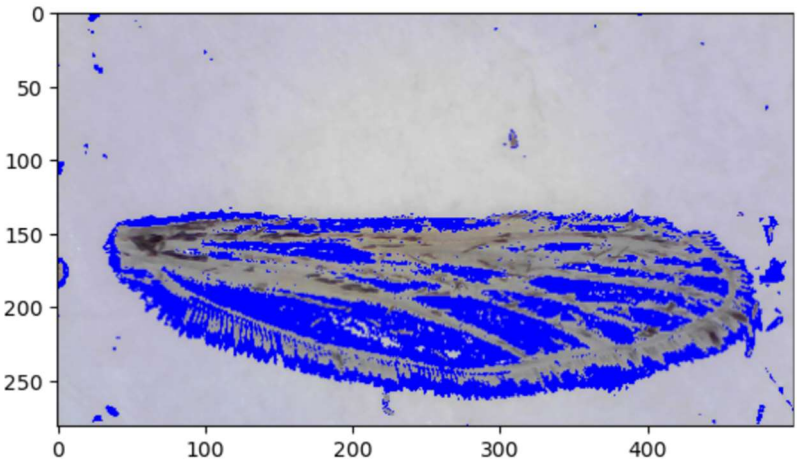


Gambar 38. Segmentasi dengan K Means dimana K = 4

Tahap 11 Membuat salinan dari image. Kemudian menyamarkan image dengan mengubah warna piksel

pada cluster yang dipilih menjadi warna biru.

```
BLUE = (0,0,255)
RED = (255,0,0)
cluster =3
masked_image = np.copy(image)
masked_image[labels_reshape == cluster] = [BLUE]
cv2.imwrite('images/maskedsayap.jpg',masked_image)

plt.imshow(masked_image);
```

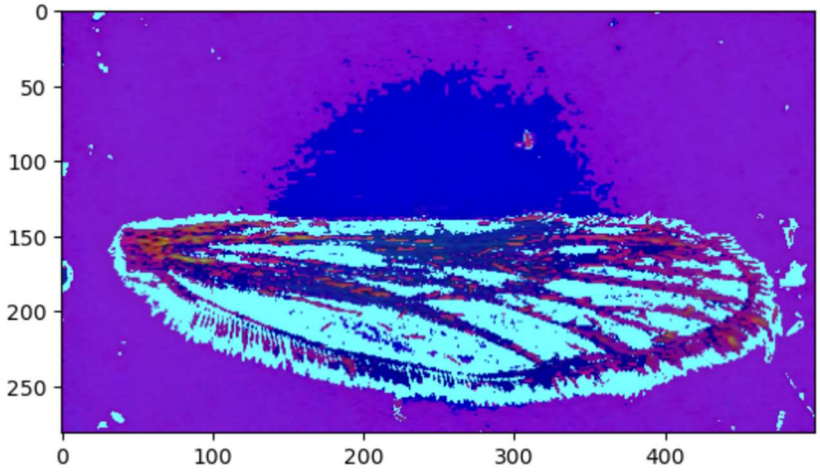


Gambar 39. Masking Image dari hasil Segmentasi

Tahap 12 Perintah cv2.COLOR_RGB2HSV digunakan mengkonversi ruang warna RGB ke ruang warna HSV

```
hsv_img = cv2.cvtColor(masked_image, cv2.COLOR_RGB2HSV)
plt.imshow(hsv_img)
```

```
<matplotlib.image.AxesImage at 0x174200f4b10>
```



Gambar 40. Image dengan Ruang Warna HSV

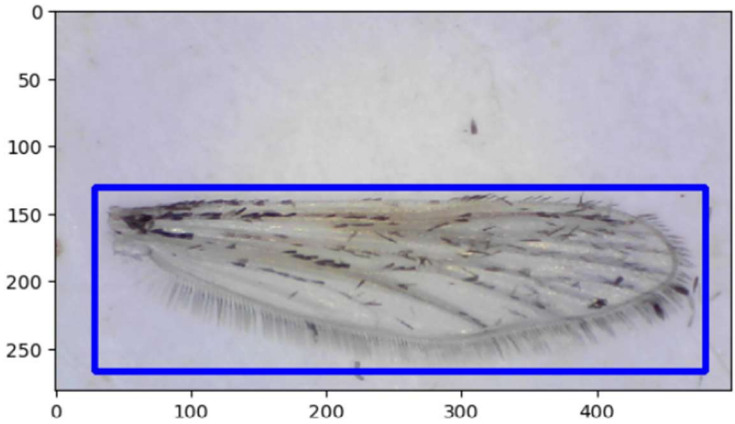
Tahap 13 : Perintah ini digunakan untuk melihat nilai hue saturation value (HSV)

```
blue = np.uint8([[[255,0,0]]])
hsv_blue = cv2.cvtColor(blue,cv2.COLOR_BGR2HSV)
print(hsv_blue)
```

```
[[[120 255 255]]]
```

Tahap 14: Melakukan deteksi kontur dan membuat *bounding box* pada area kontur

```python
# Assuming hsv_img, image, and other variables are defined

lower_blue = (120, 250, 100)
upper_blue = (255, 255, 255)
COLOR_MIN = np.array([lower_blue], np.uint8)
COLOR_MAX = np.array([upper_blue], np.uint8)

frame_threshed = cv2.inRange(hsv_img, COLOR_MIN, COLOR_MAX)

ret, thresh = cv2.threshold(frame_threshed, 127, 255, 0)
contours, hierarchy = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

# Check if contours are found
if contours:
    areas = [cv2.contourArea(c) for c in contours]
    max_index = np.argmax(areas)
    cnt = contours[max_index]

    x, y, w, h = cv2.boundingRect(cnt)

    pad_w = 2
    pad_h = 3
    pad_x = 2
    pad_y = 3

    cv2.rectangle(image, (x - pad_x, y - pad_y), (x + w + pad_w, y + h + pad_h), (255, 0, 0), 3)

    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
    plt.show()
else:
    print("No contours found.")
```

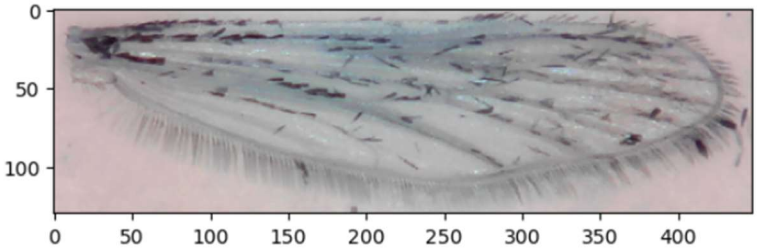


Gambar 41. Hasil Bounding Box Pada Image

Tahap 15. Jika ingin memotong image berdasarkan bounding box

```
crop=image[y:y+h,x:x+w]
plt.imshow(crop)
```

```
<matplotlib.image.AxesImage at 0x14dc4445010>
```



Gambar 42. Hasil Crop Berdasarkan Bounding Box

Tahap 16. Menyimpan hasil crop dari deteksi gambar yang telah dilakukan ke format *.jpg

```
# Save the segmented image
io.imsave('segmented_image.jpg', (crop).astype(np.uint8))
```

DAFTAR PUSTAKA

Aladžuz, A., Delalic, A., & Sceta, L. (2022). *Cluster Analysis in Python: An Example of Market Segmentation* (pp. 1032–1041). https://doi.org/10.1007/978-3-031-05230-9_122

Carsten Steger, Markus Ulrich, C. W. (2018). *Machine Vision Algorithms and Applications, 2nd Edition*. 516. https://www.wiley.com/en-in/Machine+Vision+Algorithms+and+Applications%2C+2nd+Edition-p-9783527413652

Dhabliya, D., Ugli, I. S. M., Murali, M. J., Abbas, A. H. R., & Gulbahor, U. (2023). Computer Vision: Advances in Image and Video Analysis. *E3S Web of Conferences*, *399*. https://doi.org/10.1051/e3sconf/202339904045

Dhanachandra, N., Manglem, K., & Chanu, Y. J. (2015). Image Segmentation Using K-means Clustering Algorithm and Subtractive Clustering Algorithm. *Procedia Computer Science*, *54*, 764–771. https://doi.org/10.1016/j.procs.2015.06.090

Khan, A. I., & Al-Habsi, S. (2020). Machine Learning in Computer Vision. *Procedia Computer Science*, *167*(2019), 1444–1451. https://doi.org/10.1016/j.procs.2020.03.355

Pradhana, H. W. (2014). *Visual Object Traclking using Particle Clustering Estimating Object Location using Low Cost Bearing-Only Vision*. 119–123.

OpenCV. 21 Juni 2024. K-Means Clustering in OpenCV. Diakses pada 22 Juni 2024, dari https://docs.opencv.org/4.x/d1/d5c/tutorial_py_kmeans_opencv.html

Puru Behl. Object detection using K Means. 20 Juni 2024. Diakses pada 22 Juni 2024 dari

https://www.kaggle.com/code/accountstatus/object-detection-using-k-means

Tommy, T., Siregar, R., Elhanafi, A. M., & Lubis, I. (2021). Implementasi Color Quantization pada Kompresi Citra Digital dengan Menggunakan Model Clustering Berdasarkan Nilai Max Variance pada Ruang Warna RGB. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *8*(6), 1099. https://doi.org/10.25126/jtiik.2021863490

Umargono, E., Suseno, J. E., & S. K., V. G. (2020). *K-Means Clustering Optimization using the Elbow Method and Early Centroid Determination Based-on Mean and Median*. *474*(Isstec 2019), 234–240. https://doi.org/10.5220/0009908402340240

BIODATA PENULIS

Nandang Gunawan Tunggal Waras, S.Si, M.T.
Dosen Program Studi Metrologi dan Instrumentasi
Akademi Metrologi dan Instrumentasi

Penulis adalah dosen tetap pada Program Studi Metrologi dan Instrumentasi Akademi Metrologi dan Instrumentasi. Menyelesaikan pendidikan S1 pada Jurusan Ilmu Komputer Universitas Padjadjaran dan melanjutkan S2 Program Magister Informatika di Institut Teknologi Bandung.Di kampus tersebut, penulis mengampu mata kuliah Dasar Pemrograman, Teknik Digital, Pengantar Metrologi dan lain-lain. Penulis dapat dihubungi melalui e-mail: nandanggun@gmail.com

BIODATA PENULIS

Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM.
Dosen Program Studi Teknik Informatika
Fakultas Teknologi Informasi Universitas Atma Jaya Makassar

Lahir di Cirebon Jawa Barat, tanggal 7 bulan Juni tahun 1975. Memiliki Jabatan Fungsional Lektor Kepala, Pembina Tingkat I (IV/b). Berpendidikan Sarjana Komputer (S.Kom.) di Universitas Teknologi Digital Indonesia (UTDI) tahun 1998, Magister Manajemen (M.M.) di Universitas Hasanuddin (UNHAS) tahun 2004, Magister Teknologi Informasi (M.T.) di Universitas Gadjah Mada (UGM) tahun 2007, Insinyur (Ir.) di Pendidikan Profesi Insinyur UNHAS tahun 2020, Insinyur Profesional Madya (IPM.) di Persatuan Insinyur Indonesia (PII) tahun 2021, Doktor (Dr.) di Fakultas Teknik UNHAS tahun 2023, dan Program Pendidikan Reguler Angkatan (PPRA) LX Lemhannas RI tahun 2020. Menjadi tenaga pengajar (Dosen) pada Program Studi Teknik Informatika Fakultas Teknologi Informasi Universitas Atma Jaya Makassar (UAJM). Peraih Poster terbaik DPRM Dikti tahun 2016. Dosen berprestasi IKDKI tahun 2020 dan 2021. Pernah menjabat Kepala UPT

Komputer, Kepala BAPSI, Wakil Dekan FT, Dekan FT dan FTI, Wakil Rektor III, Ketua Penjaminan Mutu. Tim PAK Dosen dan Asesor BKD UAJM. Reviewer International Conference dan Jurnal SINTA. Pemenang Hibah Kemdikbud Penelitian Dosen Pemula, Bersaing, Fundamental, dan Strategi Nasional. Penulis artikel media massa Tribun Timur, Koinonia, Bisnis Sulawesi, Hidup KatolikCom, Mirifica.net, OMKNet, KatolikanaTV, Jalan Hidup Katolik, dll. Penulis Buku di Kanisius, Sada Kurnia Pustaka, Aksara Sastra Media, Future Science, HEI Publishing, Mifandi Mandiri Digital, Rey Media Grafika, Widina Salemba, dan Cendikia Mulia Mandiri. Aktifis organisasi IKA Sesawi.net, Lemhannas RI LX, IARMI, DPP ISKA, APTIKOM, BAPOMI Sulsel, LP3KD Sulsel, IKDKI SulSelTraBar, Komkep KAMS, Komsos KAMS, PUKAT KAMS, TPP KAMS, FMKI KAMS, UPS KAMS, Pengurus Kebun Sawit Laimbo, FDI, PII Makassar, INAPR, Dewan Keuangan Paroki dan Program Ayo Sekolah Mariso, dll.

Penulis dapat dihubungi melalui e-mail: ntsaptadi@gmail.com

BIODATA PENULIS

Ayunda Kusuma Wardani, S.Tr.Kom
Program Studi Teknik Informatika
Politeknik Negeri Jember

Penulis lahir di Jember tanggal 12 April 2002. Penulis adalah lulusan pada Program Studi Teknik Informatika (TIF) Jurusan Teknologi Informasi Politeknik Negeri Jember. Selama menempuh pendidikan di Program Studi TIF penulis berminat mempelajari Kecerdasan Buatan. Penulis juga pernah mendapatkan pendanaan Program Kreativitas Mahasiswa bidang Karsa Cipta dengan judul ”Inovasi Liquid Filling Machine dengan Digital Twin Guna Memudahkan Proses Monitoring dan Efisiensi Produksi Pengalengan Ikan’ Penulis dapat dihubungi melalui e-mail: ayundakusumawardani12@gmail.com

BIODATA PENULIS

Victor Benny Alexsius Pardosi, S.Kom., M.Sc.

Fakultas Ilmu Komputer, Universitas Dharma AUB Surakarta

PT Transformasi Data Digital (HostData.id)

Penulis lahir di Aceh Tengah pada tanggal 31 Januari 1991. Merupakan seorang praktisi IT yang memiliki lebih dari sepuluh tahun pengalaman sebagai Web Developer dan System Administrator, selain itu menekuni bidang bisnis digital mulai dari periklanan sampai pengoptimalan SEO yang diterapkan pada produk dan layanan sendiri. Penulis juga mengabdi sebagai dosen di Program Studi Sistem Informasi di Fakultas Ilmu Komputer, Universitas Dharma AUB Surakarta. Menyelesaikan pendidikan S1 pada Jurusan Sistem Informasi di STMIK Dharmapala Riau lalu melanjutkan S2 jurusan Computer Science di Tomsk Polytechnic University, Russia. Tertarik dalam penelitian bidang Business Digital, Data Mining, Cyber Security, Data Protection dan Artificial Intelligence. Ketika buku ini ditulis, masih aktif terlibat sebagai Web Developer di PT Transformasi Data Digital dan sebagai SysAdmin di HostData.id.

BIODATA PENULIS

Qonitatul Hasanah, S.S.T., M.Tr.T.
Dosen Program Studi Teknik Informatika PSDKU Nganjuk
Jurusan Teknologi Informasi
Politeknik Negeri Jember

Penulis lahir di Malang pada tanggal 09 Mei 1994. Penulis adalah dosen tetap pada Program Studi Teknik Informatika PSDKU Nganjuk, Jurusan Teknologi Informasi, Politeknik Negeri Jember. Penulis menyelesaikan pendidikan D4 pada Program Studi Teknik Informatika dan melanjutkan pendidikan S2 pada Program Studi Magister Terapan Teknik Elektro dengan Konsentrasi Ilmu Komputer. Bidang keilmuan penulis adalah pemrosesan citra digital (digital image processing), namun baru-baru ini penulis juga tertarik mendalami ilmu Data Science. Penulis telah berkontribusi dalam berbagai penelitian dan publikasi ilmiah, serta aktif mengikuti seminar dan workshop di bidang keahliannya. Selain mengajar, penulis juga terlibat dalam berbagai proyek pengabdian masyarakat yang berhubungan dengan teknologi informasi dan komunikasi, dengan fokus pada penerapan teknologi untuk

meningkatkan kualitas hidup masyarakat.Penulis dapat dihubungi melalui e-mail: qonitatul@polije.ac.id

BIODATA PENULIS

Arvita Agus Kurniasari, S.ST.,M.Tr.Kom.
Dosen Program Studi Teknik Informatika
Jurusan Teknologi Informasi Politeknik Negeri Jember

Penulis lahir di Surabaya tanggal 31 Agustus 1993. Penulis adalah dosen tetap pada Program Studi Teknik Informatika (D4), Jurusan Teknologi Informasi Politeknik Negeri Jember. Penulis menyelesaikan pendidikan Sarjana Sain Terapan (D4) di Politeknik Elektrinonika Negeri Surabaya Jurusan Teknologi Informasi dan melanjutkan Magister (S2) di Program Pascasarjana Politeknik Elektrinonika Negeri Surabaya.Penulis dapat dihubungi melalui e-mail: arvita@polije.com

BIODATA PENULIS

Muhammad Hafidh Firmansyah, S.Tr.Kom., M.Sc.
Dosen Program Studi Teknik Komputer
Jurusan Teknologi Informasi Politeknik Negeri Jember

Penulis lahir di Bondowoso tanggal 14 Februari 1997. Penulis adalah dosen tetap pada Program Studi Teknik Komputer Jurusan Teknologi Informasi Politeknik Negeri Jember. Menyelesaikan pendidikan S1 pada Prodi Teknik Informatika Politeknik Negeri Jember dan melanjutkan S2 pada Jurusan Computer Science and Engineering Kyungpook National University. Saat ini penulis aktif melakukan kegiatan penelitian dan pengajaran pada bidang Kecerdasan Buatan, IoT serta aplikasi berbasiskan Komputasi Cerdas.Penulis dapat dihubungi melalui e-mail: hafidh@polije.ac.id

BIODATA PENULIS

Aji Seto Arifianto, S.ST, M.T
Dosen Program Studi Teknik Informatika
Jurusan Teknologi Informasi Politeknik Negeri Jember

Penulis lahir di Jember tanggal 28 November 1985. Penulis menyelesaikan pendidikan Sarjana Sain Terapan (D4) di Politeknik Elektrinonika Negeri Surabaya dan melanjutkan Magister (S2) di Program Pascasarjana Universitas Brawijaya. Penulis mengampu mata kuliah *Artificial Intelligence, Computer Vision, Decision Support System*. Penulis juga tertarik untuk mengembangkan aplikasi dengan *Immersive Technology* . Selain Buku ini, penulis juga terlibat dalam kolaborasi menulis Buku "10 Metode SPK Favorit Di Masa Depan" dan "Development of Artificial Intelligence Applications". Penulis dapat dihubungi melalui e-mail: ajiseto@polije.ac.id

BIODATA PENULIS

Giandari Maulani, S.Kom, M.Kom
Kaprodi STIE Putra Perdana Indonesia (PPI)
Tangerang-Indonesia

Penulis menyelesaikan pendidikan Strata Satu (S1) pada Universitas Raharja Tangerang jurusan Sistem Informasi dan Strata Dua (S2) pada Universitas Budi Luhur Jakarta dengan jurusan Teknologi Sistem Informasi. Saat ini Penulis bekerja sebagai Kaprodi (Ketua Program Studi) dan Dosen Tetap (berstatus Sertifikasi Dosen) pada STIE Putra Perdana Indonesia (PPI) di Tangerang dan memiliki 2 (dua) Pekerjaan Online lainnya. Penulis memiliki pengalaman menulis antara lain: Pernah mendapatkan Hibah Pendidikan Jarak Jauh (PJJ) dari Kementerian Pendidikan dan Kebudayaan Republik Indonesia di tahun 2021 senilai Rp. 49.000.000,- dan pernah mendapatkan Juara III tingkat Nasional Lomba Karya Tulis Inovatif (LKTI) Bidang Pemerintahan di tahun 2021. Saat ini Penulis memiliki 64 (enam puluh empat) Paper yang telah terpublikasi pada berbagai Jurnal Nasional dengan 5 (Lima)

International Journal terindeks Scopus selama rentang tahun 2015-2023.

Mulai September 2023 sampai dengan Sekarang, Penulis aktif membuat Buku Kolaborasi dalam bidang Komputer, bidang Pendidikan, bidang Manajemen, bidang Ekonomi, dll. Buku-buku Kolaborasi tersebut antara lain :

- Buku Seni dan Sains CNC DIY: "Jembatan Kreativitas dan Teknologi Mesin".
- Buku Pendidikan Inklusi.
- Buku Manajemen Mutu Pendidikan.
- Buku *Micro Teaching* (Teoritis & Praktis).
- Buku Pendidikan Anak Usia Dini.
- Buku Pendidikan Multikultural.
- Buku Bidang IT Fundamental Algoritma.
- Buku Komunikasi Pendidikan.
- Buku Revolusi Pendidikan Merdeka Belajar Kampus Merdeka /MBKM.
- Buku Manajemen Strategi Menghadapi Industri 5.0
- Buku Analisa Sistem.
- Buku Pemanfaatan dan Penerapan *Internet of Things*/IoT di Berbagai Bidang.
- Buku Komputer dan Masyarakat.
- Buku Interaksi Manusia & Komputer.
- Buku *Development of Artificial Intelligence Applications.*
- Buku Penerapan *Data Mining* di berbagai Bidang.
- Buku Rekayasa Perangkat Lunak.
- Buku Konsep Dasar Bisnis internasional.
- Buku Manajemen Pelayanan Publik.
- Buku Transformasi Metodologi Pembelajaran.
- Buku Manajemen Pendidikan Tinggi.
- Buku Kebijakan Pendidikan.
- Buku Pusat Kegiatan Belajar Masyarakat (PKBM)

- Buku Revitalisasi Ekonomi Sumber Daya Alam dan Lingkungan di Era 5.0.
- Buku *Exploring The Power of Big Data and Data Analytics for Tomorrow's Insight*.
- Buku Pendidikan Kewarganegaraan.
- Buku Strategi Pembelajaran Anak Usia Dini.
- Buku Manajemen Keuangan Internasional.
- Buku Pemanfaatan Teknologi Informasi pada Masa Society 5.0.
- Buku Dunia Multimedia.
- Buku ICT dan Multimedia.
- Buku Kuasai Machine Learning & Computer Vision dalam Sekejap.
- Buku Kecerdasan Buatan.
- Buku Strategi Inovatif dalam Manajemen Bisnis.
- Buku Penerapan & Implementasi Big Data di berbagai Sektor.
- Buku Perilaku Organisasi.
- Buku Sistem Informasi Manajemen.
- Buku Pengenalan dan Dasar Teknik Robotika.
- Buku Evaluasi Pembelajaran.
- Buku Pengembangan Kurikulum: Teori, Model dan Praktik.
- Buku Augmented and Virtual Reality.

Email Penulis: maulanigiandari@gmail.com

BIODATA PENULIS

Johar Nur Iin, S. Kom., MT
Dosen Program Studi Sistem Informasi
Fakultas Teknik Informatika
Universitas Sembilanbelas November Kolaka

Penulis lahir di Makassar tanggal 10 Desember 1986. Penulis adalah dosen tetap di Fakultas Teknologi Informasi Universitas Sembilanbelas November Kolaka. Kolaka. Indonesia. Menyelesaikan pendidikan S1 pada Sistem Informasi STMIK Dipanegara Makasar sekarang Universitas Dipanegara Makassar dan melanjutkan S2 pada Jurusan Teknik Elektro Konsentrasi Teknik Informatika Universitas Hasanuddin.