**DETEKSI PENGEMUDI KANTUK MENGGUNAKAN ALGORITMA YOLOv8**

**TUGAS AKHIR**

**Text

Description automatically generated**

**OLEH:**

**MUHAMAD ARIE SETYA PUTRA PALA**

**1922021**

**PROGRAM STUDI TEKNIK KOMPUTER**

**FAKULTAS TEKNOLOGI INFORMASI**

**INSTITUT TEKNOLOGI BATAM**

**2023**

**DETEKSI PENGEMUDI KANTUK MENGGUNAKAN ALGORITMA YOLOv8**

**TUGAS AKHIR**

**DIAJUKAN SEBAGAI SALAH SATU SYARAT UNTUK MENDAPATKAN GELAR SARJANA (STRATA-1) PADA PROGRAM STUDI TEKNIK KOMPUTER**

**Text

Description automatically generated**

**PROGRAM STUDI TEKNIK KOMPUTER**

**FAKULTAS TEKNOLOGI INFORMASI**

**INSTITUT TEKNOLOGI BATAM**

**2023**

# DAFTAR ISI

[DAFTAR ISI iii](#_Toc169522928)

[DAFTAR GAMBAR v](#_Toc169522929)

[DAFTAR TABEL vi](#_Toc169522930)

[BAB 1 PENDAHULUAN 1](#_Toc169522931)

[1.1 Latar Belakang 1](#_Toc169522932)

[1.2 Perumusan Masalah 3](#_Toc169522933)

[1.3 Batasan Masalah 3](#_Toc169522934)

[1.4 Tujuan dan Manfaat Penelitian 3](#_Toc169522935)

[1.4.1 Tujuan 3](#_Toc169522936)

[1.4.2 Manfaat 4](#_Toc169522937)

[1.5 Sistematika Penulisan 6](#_Toc169522938)

[BAB 2 TINJAUAN PUSTAKA 7](#_Toc169522939)

[2.1 YOLOV8 7](#_Toc169522940)

[2.1.1 Darknet 9](#_Toc169522941)

[2.1.2 Pytorch 10](#_Toc169522942)

[*2.2* *FACE DETECTION* 12](#_Toc169522943)

[*2.3* *APPLE CORE ML* 13](#_Toc169522944)

[2.4 Convolutional Neural Networks 14](#_Toc169522945)

[2.4.1 Convolutional Layer 14](#_Toc169522946)

[2.4.2 Pooling Layer 15](#_Toc169522947)

[2.4.3 Fully Connected Layer 15](#_Toc169522948)

[2.4.4 Activation Function 15](#_Toc169522949)

[2.5 DEEP LEARNING 16](#_Toc169522950)

[*2.6* *DATASET* 17](#_Toc169522951)

[2.7 *Python* 19](#_Toc169522952)

[2.8 Penelitian Terdahulu 21](#_Toc169522953)

[BAB 3 Metodologi Penelitian 31](#_Toc169522954)

[3.1 Metodologi Penelitian 31](#_Toc169522955)

[3.1.1 Metode Kuantitatif 31](#_Toc169522956)

[3.2 Kerangka Penelitian 32](#_Toc169522957)

[3.2.1 Identifikasi Masalah 32](#_Toc169522958)

[3.2.2 Studi Literatur 34](#_Toc169522959)

[3.2.3 Pengumpulan Data 35](#_Toc169522960)

[3.2.4 Pengolahan Data 36](#_Toc169522961)

[3.2.5 Analisa Hasil 36](#_Toc169522962)

[BAB 4 Analisa Perancangan 39](#_Toc169522963)

[4.1 Analisa Sistem 39](#_Toc169522964)

[4.2 Desain Sistem 39](#_Toc169522965)

[4.2.1 Persiapan Data 39](#_Toc169522966)

[4.2.2 Pelatihan Model 39](#_Toc169522967)

[4.3 Evaluasi Model 39](#_Toc169522968)

[4.4 Pengujian Model 39](#_Toc169522969)

[4.5 Konversi Model 39](#_Toc169522970)

[BAB 5 Hasil dan Pembahasan 40](#_Toc169522971)

[BAB 6 Kesimpulan dan Saran 42](#_Toc169522972)

[6.1 Kesimpulan 42](#_Toc169522973)

[6.2 Saran 42](#_Toc169522974)

[DAFTAR PUSTAKA 44](#_Toc169522975)

# DAFTAR GAMBAR

[Gambar 1. 1 Gambar Diagram persentase pertumbuhan kecelakaan berdasarkan waktu 1](#_Toc169031488)

Gambar 2. 1 Gambar ilustrasi YOLOv8 (Sumber : https://www.ultralytics.com) 7

[Gambar 2. 2 Gambar ilustrasi deteksi wajah (Sumber : https://studentsforhumanity.it) 12](#_Toc169031704)

[Gambar 2. 3 Gambar ilustrasi Apple core ML (Sumber : https://developer.apple.com/machine-learning/core-ml/) 13](#_Toc169031705)

[Gambar 2. 4 Gambar ilustrasi algoritma CNN (Sumber : https://www.researchgate.net) 14](#_Toc169031706)

[Gambar 2. 5 Gambar ilustrasi tahapan pada deep learning (Sumber : https://www.researchgate.net) 16](#_Toc169031707)

[Gambar 2. 6 Gambar ilustrasi dataset gambar (Sumber : https://neurohive.io) 17](#_Toc169031708)

[Gambar 2. 7 Gambar Logo Python (Sumber : https://www.ntuclearninghub.com) 19](#_Toc169031709)

[Gambar 3. 1 Gambar kerangka penelitian 32](#_Toc169031710)

# DAFTAR TABEL

[Tabel 2. 1 Tabel Variasi dari YOLO 9](#_Toc169031794)

[Tabel 2. 2 Tabel penelitian terdahulu 21](#_Toc169031795)

[Tabel 3. 1 Tabel perbedaan hasil akurasi 36](#_Toc169031800)

# PENDAHULUAN

## Latar Belakang

Mengemudi adalah aktivitas penting dan rutin bagi banyak orang, sehingga diperlukan upaya mendalam untuk memahami, mengenali, dan memprediksi perilaku mengemudi mereka[1]. Setiap pengemudi memiliki teknik, pengetahuan, dan perasaan yang berbeda karena karakteristik mengemudi yang unik. Faktor-faktor manusia seperti kurangnya perhatian, gangguan kognitif, dan pandangan yang tidak tepat sering kali menyebabkan tindakan pengemudi yang tidak memadai[1]. Misalnya, pengemudi mungkin tidak memperhatikan lalu lintas dan rangsangan di lingkungan mengemudi, termasuk jalan dan kendaraan lain, atau hanya memperhatikan dalam waktu singkat. Hal ini meningkatkan risiko terjadinya kecelakaan[2].

Salah satu masalah serius yang mengancam keselamatan di jalan adalah kantuk pada pengemudi. Kantuk tidak hanya mengurangi kemampuan pengemudi untuk bereaksi dengan cepat, tetapi juga meningkatkan kemungkinan terjadinya kesalahan yang bisa berakibat fatal. Oleh karena itu, sangat penting untuk merancang algoritma deteksi kantuk yang efektif untuk mencegah kecelakaan[3]. Algoritma ini harus mampu mendeteksi tanda-tanda kantuk secara akurat dan memberikan peringatan dini kepada pengemudi, sehingga tindakan pencegahan dapat diambil sebelum kecelakaan terjadi.

Gambar 1. 1 Gambar Diagram persentase pertumbuhan kecelakaan berdasarkan waktu

Kantuk pada pengemudi berkontribusi pada banyak kecelakaan lalu lintas di seluruh dunia[3]. Berdasarkan informasi dari Korlantas Polri, tercatat sebanyak 103.645 kecelakaan terjadi dalam satu tahun. Yang lebih mengkhawatirkan, terdapat peningkatan jumlah kecelakaan sebesar 30% pada malam hari dibandingkan dengan tahun sebelumnya. Peningkatan ini menunjukkan bahwa kantuk dan kurangnya perhatian pengemudi merupakan faktor utama yang mempengaruhi keselamatan berkendara, terutama pada malam hari ketika tubuh manusia secara alami cenderung merasa lebih lelah.

Serta menurut *Traffic Safety Administration* (NHTSA) pada tahun 2017, melaporkan 91.000 kecelakaan kendaraan yang melibatkan pengemudi yang mengantuk. Selain kantuk, mengemudi yang terganggu juga berisiko, mengakibatkan 3.142 kematian pada tahun 2020 [4], dan 1,3 juta orang meninggal setiap tahun dalam kecelakaan lalu lintas, menjadikannya penyebab kematian ketujuh di seluruh dunia di antara orang muda berusia antara 15 dan 29 tahun[4].

Untuk mengatasi masalah ini, diperlukan teknologi deteksi kantuk pengemudi yang andal dan efektif. Algoritma deteksi kantuk yang dirancang dengan baik dapat membantu mengidentifikasi tanda-tanda kantuk pada pengemudi secara dini, seperti mata yang sering berkedip, kepala yang menunduk, atau wajah yang tampak lelah. Dengan data ini, sistem dapat memberikan peringatan kepada pengemudi untuk beristirahat atau mengambil tindakan lain yang diperlukan untuk memastikan keselamatan berkendara.

Mengembangkan teknologi semacam ini memerlukan pendekatan berbasis data yang komprehensif. Data dapat dikumpulkan dari berbagai sumber, termasuk kamera yang merekam wajah pengemudi dan sensor yang memantau kondisi fisik mereka. Selanjutnya, data ini dapat dianalisis menggunakan algoritma pembelajaran mesin seperti YOLOv8, yang mampu mengenali pola dan tanda-tanda kantuk dengan tingkat akurasi yang tinggi.

Dengan memahami dan mengatasi faktor-faktor manusia yang berkontribusi pada kecelakaan lalu lintas, kita dapat meningkatkan keselamatan jalan raya secara signifikan. Penggunaan teknologi deteksi kantuk yang canggih adalah langkah penting menuju pengurangan jumlah kecelakaan dan korban jiwa di jalan. Investasi dalam penelitian dan pengembangan teknologi ini tidak hanya akan menyelamatkan nyawa, tetapi juga mengurangi kerugian ekonomi yang disebabkan oleh kecelakaan lalu lintas.

Oleh karena itu, skripsi ini sangat relevan dan layak dilakukan karena memiliki potensi besar untuk memberikan kontribusi signifikan dalam meningkatkan keselamatan berkendara. Dengan mengembangkan dan menerapkan algoritma deteksi kantuk yang efektif, penelitian ini dapat menjadi langkah penting dalam upaya mengurangi kecelakaan lalu lintas dan menyelamatkan banyak nyawa di jalan raya. Penelitian ini juga dapat membuka jalan bagi inovasi teknologi di masa depan, memperkuat keselamatan pengemudi, dan meningkatkan kualitas hidup masyarakat.

## Perumusan Masalah

Adapun beberapa perumusan masalah berdasarkan latar belakang yang telah penulis sampaikan adalah sebagai berikut:

1. Bagaimana efektivitas dan efisiensi implementasi Mengenal objek menggunakan metode YOLO dalam situasi di mana koneksi internet mungkin terbatas atau tidak stabil?
2. Apa tantangan teknis utama yang dihadapi dalam mengembangkan dan menjalankan Mengenal objek menggunakan metode YOLO?
3. Bagaimana tingkat kualitas dan keakuratan dari respons Mengenal objek menggunakan metode YOLO dibandingkan dengan implementasi metode lain?

## Batasan Masalah

1. Penelitian ini hanya mencakup pengembangan, pelatihan, dan evaluasi model deteksi kantuk menggunakan algoritma YOLOv8
2. Penelitian ini hanya mencakup proses konversi model machine learning ke format .mlmodel, tanpa menyentuh aspek lain dari pengembangan model.

## Tujuan dan Manfaat Penelitian

### Tujuan

Tujuan dari penelitian ini:

1. Mengenali bagaimana cara kerja metode *YOLO* yang popular terhadap topik mengenal objek
2. Untuk menguji dan mengevaluasi model YOLO yang diintegrasikan dengan Apple Core ML. Hal ini melibatkan pengukuran kinerja model dalam hal akurasi, kecepatan, dan efisiensi.
3. Dapat mengembangkan API dari sistem *Mengenal objek menggunakan metode YOLO* yang sudah berjalan secara lokal untuk kebutuhan lainnya bahkan untuk dikomersialkan.
4. Dapat mengembangkan sebuah aplikasi iOS yang mampu mengenali objek secara *real-time* menggunakan kamera perangkat iOS bertujuan untuk kemudahan dalam penggunaan aplikasi dan memberikan pengenalan objek yang akurat.
5. Untuk mengintegrasikan model deteksi objek YOLO dengan *Apple Core ML* yang melibatkan pelatihan model YOLO dan konversi model tersebut agar dapat dijalankan di platform iOS dengan bantuan Core ML.

### Manfaat

Adapun beberapa manfaat dari penelitian bagi penulis adalah sebagai berikut:

1. Pemahaman Mendalam tentang Teknologi: Peneliti akan memperoleh pengetahuan yang lebih baik tentang integrasi algoritma YOLOv8 dengan Apple Core ML, serta cara memanfaatkan kemampuan kedua teknologi ini untuk mendeteksi kantuk pada pengemudi.
2. Pengembangan Keterampilan Teknis: Penelitian ini akan membantu peneliti meningkatkan keterampilan dalam machine learning dan computer vision, serta penguasaan alat-alat dan framework yang relevan seperti PyTorch, Darknet, dan Core ML.
3. Kontribusi pada Pengetahuan Akademis: Hasil penelitian ini dapat menambah literatur yang ada dalam bidang deteksi objek dan machine learning, memberikan wawasan baru bagi peneliti lain yang tertarik dengan topik serupa.
4. Dasar untuk Penelitian Selanjutnya: Temuan dari penelitian ini dapat menjadi dasar bagi penelitian lanjutan, baik oleh peneliti yang sama atau oleh peneliti lain yang ingin mengembangkan lebih jauh.
5. Pengalaman Praktis dalam Penelitian: Peneliti akan mendapatkan pengalaman praktis dalam merancang, mengimplementasikan, dan menguji model machine learning, yang dapat berguna untuk proyek-proyek masa depan.

Adapun beberapa manfaat dari penelitian bagi pendidikan adalah sebagai berikut:

1. Peningkatan Kompetensi Mahasiswa: Mahasiswa yang terlibat dalam penelitian ini akan mendapatkan pengalaman praktis dalam menerapkan teknik machine learning dan deep learning, yang sangat dihargai dalam industri teknologi. Keterlibatan langsung dalam proyek ini akan meningkatkan kompetensi teknis mereka, memperkuat keterampilan analitis, dan mempersiapkan mereka untuk tantangan yang akan mereka hadapi di tempat kerja.
2. Inspirasi untuk Penelitian Lebih Lanjut: Hasil dari penelitian ini dapat menginspirasi mahasiswa dan akademisi untuk mengeksplorasi topik-topik lain dalam bidang machine learning dan computer vision. Ini dapat memicu penelitian lanjutan yang lebih mendalam, menciptakan siklus pembelajaran dan inovasi yang berkelanjutan di lingkungan akademik.
3. Pengembangan Keterampilan Interdisipliner: Penelitian ini mencakup berbagai bidang ilmu, termasuk teknik komputer, psikologi (untuk memahami kantuk dan pengaruhnya pada pengemudi), dan keselamatan jalan. Mahasiswa yang terlibat akan mengembangkan keterampilan interdisipliner yang penting untuk memecahkan masalah kompleks di dunia nyata.

Adapun beberapa manfaat dari penelitian bagi masyarakat adalah sebagai berikut:

1. Peningkatan Keselamatan Berkendara: Dengan mengembangkan teknologi deteksi kantuk pengemudi yang akurat, penelitian ini dapat berkontribusi secara signifikan terhadap peningkatan keselamatan berkendara. Sistem deteksi kantuk yang efektif dapat memperingatkan pengemudi yang mengantuk, mengurangi risiko kecelakaan lalu lintas yang disebabkan oleh kantuk, dan menyelamatkan banyak nyawa.
2. Peningkatan Kualitas Hidup: Teknologi ini dapat membantu masyarakat dalam menjalani kehidupan yang lebih aman dan nyaman. Dengan adanya sistem deteksi kantuk yang diintegrasikan ke dalam kendaraan, pengemudi akan merasa lebih aman dan terjaga saat berkendara, terutama selama perjalanan panjang atau di malam hari.
3. Edukasi Masyarakat tentang Pentingnya Keselamatan: Implementasi dan publikasi hasil penelitian ini dapat meningkatkan kesadaran masyarakat tentang bahaya berkendara saat mengantuk dan pentingnya teknologi keselamatan. Edukasi ini dapat mengubah perilaku pengemudi, mendorong mereka untuk mengambil tindakan preventif seperti istirahat yang cukup sebelum berkendara.
4. Dukungan untuk Industri Otomotif: Hasil penelitian ini dapat memberikan data dan insight berharga bagi industri otomotif dalam mengembangkan sistem keselamatan yang lebih canggih. Pabrikan mobil dapat mengadopsi teknologi deteksi kantuk ini dalam produk mereka, meningkatkan fitur keselamatan kendaraan, dan memberikan nilai tambah bagi konsumen.
5. Kontribusi pada Kebijakan Keselamatan: Temuan dari penelitian ini dapat digunakan oleh pembuat kebijakan untuk mengembangkan regulasi dan standar keselamatan jalan yang lebih baik. Dengan bukti empiris yang kuat tentang efektivitas sistem deteksi kantuk, regulasi yang mendukung implementasi teknologi ini dapat diperkenalkan untuk meningkatkan keselamatan jalan raya secara keseluruhan.

## Sistematika Penulisan

Penulisan laporan tugas akhir terdiri dari 5 bab dengan penjelasan sebagai berikut:

**BAB I: PENDAHULUAN**

Berisi tentang latar belakang, rumusan dan batasan masalah, tujuan dan sistematika penulisan.

**BAB II: LANDASAN TEORI**

Berisi tentang teori-teori yang berhubungan dengan penelitian.

**BAB III: METODOLOGI**

Berisi tentang bagaimana penelitian dilakukan, mengumpulkan data, menganalisis data, dan mencapai tujuan penelitian.

**BAB IV: HASIL DAN PEMBAHASAN**

Berisi tentang hasil pengujian dan analisis sistem yang dibuat dan dibandingkan dengan dasar teori dan sistem lain yang dijadikan sebagai pembanding.

**BAB V: PENUTUP**

Bagian ini memuat kesimpulan dan saran-saran dari perancangan sistem, berikut juga keterbatasan yang ditemukan dan asumsi-asumsi dari penulis selama penelitian berlangsung.

# TINJAUAN PUSTAKA

## YOLOV8



Gambar 2. 1 Gambar ilustrasi YOLOv8 (Sumber : https://www.ultralytics.com)

YOLO (*You Only Look Once*) adalah sebuah algoritma dalam bidang pengenalan objek pada citra yang sangat populer di bidang *computer vision* dan *machine learning*. Seri algoritma YOLO adalah salah satu algoritma dengan pertumbuhan tercepat dan terbaik hingga saat ini, terutama algoritma YOLOv8 yang baru dirilis pada tahun 2023, yang telah mencapai tingkat akurasi tertinggi hingga saat ini. YOLOv8, yang diterbitkan pada tahun 2023, dirancang untuk menggabungkan keunggulan dari berbagai detektor objek *real-time*. YOLOv8 mendukung semua versi YOLO dan dapat beralih antara versi yang berbeda dengan mudah. Algoritma ini juga dapat berjalan di berbagai platform perangkat keras (CPU-GPU), memberikan fleksibilitas yang luar biasa[5].

YOLO (*You Only Look Once*), sebuah *object detection model* dan *image segmentation* yang dikembangkan oleh Joseph Redmon dan Ali Farhadi di Universitas Washington, diperkenalkan pada tahun 2015. Model ini dengan cepat menjadi populer berkat kecepatannya yang tinggi dan tingkat akurasi yang sangat baik. Joseph dan Ali memperkenalkan pendekatan baru untuk deteksi objek dengan mengubahnya menjadi masalah regresi satu langkah. Dimulai dari *pixel* gambar, metode ini memprediksi *bounding box* dan *class probabilities* secara bersamaan. Konsep *'unified’* yang digunakan memungkinkan prediksi beberapa *bounding box* dan probabilitas kelas secara simultan, sehingga meningkatkan kecepatan dan akurasi *object detection*[6]. Kehadiran YOLO memberikan dampak yang signifikan dalam meningkatkan potensi pada *computer vision*[6].

Berikut sedikit ringkasan mengenai perkembangan YOLO pada setiap versi :

1. YOLO-v1 (2015): Arsitektur pertama yang menggabungkan prediksi *bounding box* dan klasifikasi dalam satu tugas, memungkinkan deteksi objek secara lebih efisien dalam satu langkah[6].
2. YOLO-v2 (2016): Memperkenalkan perbaikan arsitektur seperti normalisasi *batch* untuk stabilitas pelatihan yang lebih baik, resolusi gambar yang lebih tinggi untuk deteksi lebih akurat, dan penggunaan kotak jangkar (*anchor boxes*) untuk menangani berbagai ukuran objek[6].
3. YOLO-v3 (2018): Dilengkapi dengan skor *objectness* untuk memperbaiki prediksi *bounding box*, penambahan koneksi untuk lapisan *backbone* yang meningkatkan deteksi, dan kemampuan untuk menghasilkan prediksi pada tiga tingkat granularitas, meningkatkan kinerja pada objek kecil[6].
4. YOLO-v4 (2020): Dikembangkan oleh tim baru, menambahkan penggabungan fitur yang lebih baik, *'bag of freebies'* yang mencakup teknik gratis untuk meningkatkan performa, dan *mish activation* yang meningkatkan kinerja jaringan[6].
5. YOLO-v5 (2020): Varian pertama tanpa makalah ilmiah, dibangun dengan *PyTorch* dan memiliki *repository* GitHub yang aktif, membuat implementasi lebih mudah diakses oleh pengembang dan peneliti[6].
6. YOLO-v6 (2021): Fokus pada efisiensi perangkat keras dengan desain ulang modul *backbone* (*EfficientRep*) dan *neck* (Rep-PAN), serta pengenalan konsep *anchor-free* dan kepala yang terpisah untuk meningkatkan kinerja keseluruhan[6].
7. YOLO-v7 (2022): Memperkenalkan reformasi arsitektur dengan mempertimbangkan memori yang diperlukan untuk menjaga lapisan dalam memori dan jarak untuk *back-propagate* gradien, serta implementasi E-ELAN untuk agregasi lapisan yang lebih baik[6].
8. YOLO-v8 (2023): Dirilis oleh Ultralytics, menunjukkan kinerja *throughput* yang menjanjikan dibandingkan dengan pendahulunya, dengan parameter komputasi yang serupa. Ini mencakup berbagai peningkatan yang akan dijelaskan lebih lanjut dalam makalah yang akan datang[6].

Berikut tabel menyajikan berbagai variasi YOLO, *framework* yang digunakan, serta komentar mengenai kelebihan dan kekurangan masing-masing variasi. Informasi ini bertujuan untuk memberikan gambaran yang lebih komprehensif mengenai evolusi YOLO dan memilih variasi yang paling sesuai dengan kebutuhan penelitian ini.

Tabel 2. 1 Tabel Variasi dari YOLO

|  |  |  |
| --- | --- | --- |
| **Variant** | **Framework** | **Comment** |
| V1 | Darknet | Hanya mendeteksi maksimal dua objek dalam grid yang sama.  Memperkenalkan norma batch, pengelompokan k-means untuk Anchor Box. |
| V2 | Darknet | Mampu mendeteksi> 9000 kategori. |
| V3 | Darknet | Memanfaatkan prediksi multi-skala dan pengumpulan piramida spasial yang menghasilkan bidang reseptif yang lebih besar. |
| V4 | Darknet | Memperkenalkan “bag of freebies” termasuk penggunaan CloU loss |
| V5 | Pytorch | Varian pertama yang berbasis di PyTorch, membuatnya lebih tersedia untuk banyak orang. Mengintegrasikan proses pemilihan anchor ke dalam pipeline YOLO-v5. |
| V6 | Pytorch | Berfokus pada pengaturan industri, memperkenalkan pipeline tanpa anchor. Memperkenalkan mekanisme penentuan loss baru (VFL, DFL, dan SIoU/GIoU). |
| V7 | Pytorch | Pengenalan arsitektural termasuk E-ELAN untuk konvergensi lebih cepat bersama dengan "bag-of-freebies" termasuk RepConvN dan perencanaan reparameterisasi. |
| V8 | Pytorch | Anchor-free, mengurangi jumlah kotak prediksi sambil mempercepat non-maximum suppression. |

Setelah melihat variasi YOLO beserta framework yang mendukungnya, penting untuk memahami lebih dalam mengenai dua framework utama yang sering digunakan dalam implementasi YOLO, yaitu Darknet dan PyTorch. Berikut ini adalah pembahasan mengenai masing-masing framework tersebut:

### Darknet

Darknet adalah model convolutional neural network (CNN) yang ringan dan telah mendapatkan perhatian di bidang deep learning dan neural networks[7]. Dengan desain yang berfokus pada efisiensi dan kinerja tinggi, Darknet unggul dalam mengelola tugas pemrosesan data yang rumit. Arsitekturnya yang sederhana menawarkan kekuatan dan fleksibilitas, membuatnya cocok untuk berbagai aplikasi. Kemampuan model ini dalam menangani dan menganalisis dataset yang besar dengan presisi menyoroti pentingnya dalam lanskap yang berpusat pada data saat ini.

Arsitektur deep learning yang canggih dari Darknet sangat mahir dalam memproses dan menafsirkan sejumlah besar data[7]. Struktur Artificial Neural Network (ANN) memungkinkan model ini untuk belajar dan mengidentifikasi fitur kompleks dalam dataset seperti gambar medis. Kemampuan ini sangat penting untuk identifikasi dan analisis anomali yang akurat, termasuk order monuments, sehingga meningkatkan akurasi diagnostik dalam pencitraan medis.

Selain itu, Darknet dipuji karena adaptabilitas dan kinerjanya dalam implementasi neural network. Sebagai sebuah open-source framework, darknet menyediakan tools yang fleksibel dan dapat diakses oleh para peneliti dan pengembang[7]. Kinerjanya yang luar biasa dalam tugas deteksi objek telah menjadikannya salah satu jaringan deep learning terkemuka yang tersedia saat ini.

Selain itu, framework Darknet mendukung akselerasi GPU selama fase pelatihan[8]. Kemampuan ini secara signifikan mengurangi waktu pelatihan dibandingkan dengan penggunaan CPU, membuat pelatihan model lebih efisien dan memfasilitasi penerapan solusi neural network dengan lebih cepat. Singkatnya, kombinasi desain Darknet yang ringan, kapabilitas ANN yang kuat, aksesibilitas open-source, dan akselerasi GPU menjadikannya alat yang penting di bidang deep learning. Efektivitasnya dalam pencitraan medis dan deteksi objek menunjukkan kemampuannya untuk menangani tantangan kompleks yang intensif data dengan efisien.

### Pytorch

Seiring dengan semakin besarnya dan kompleksnya model machine learning (ML), alat dan framework yang digunakan untuk mengembangkannya juga telah beradaptasi. Sebelumnya, model ML digambarkan melalui grafik aliran data statis, tetapi sifat dinamis model modern sekarang memerlukan pendekatan yang lebih fleksibel. Evolusi ini telah mengarah pada penggunaan luas framework eager-mode seperti PyTorch, yang telah menjadi standar untuk menciptakan model ML modern[9].

Saat ini, sebagian besar model ML dikembangkan dalam framework ML, sering kali menggunakan library Python karena kemudahan penggunaannya dan kapabilitasnya yang kuat [9]. PyTorch, khususnya, telah mendapatkan popularitas yang signifikan di kalangan praktisi dan peneliti deep learning. Faktor utama yang meningkatkan popularitasnya adalah kinerja PyTorch yang secara signifikan ditingkatkan oleh eksekusi GPU asinkron, sebuah fitur kunci. Ini berarti bahwa sementara kode Python berjalan di CPU, operasi tensor dijalankan secara bersamaan di GPU. PyTorch kemudian mengembalikan kendali ke Python bahkan sebelum operasi tersebut selesai. Ini memungkinkan Python untuk melanjutkan tugasnya sementara PyTorch mengantri operasi di CUDA, secara efektif menyembunyikan latensi yang signifikan yang muncul dalam komunikasi CPU-GPU[9].

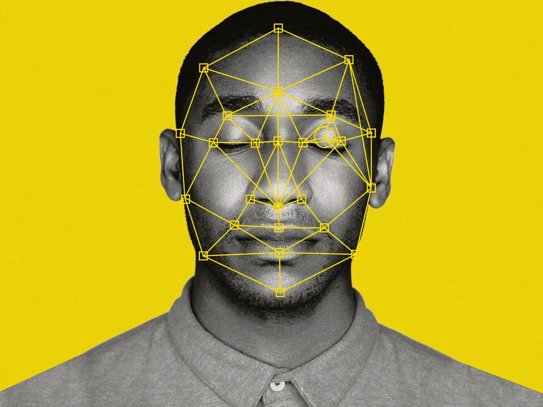
Untuk lebih mengoptimalkan kinerja, PyTorch menggunakan allocator caching CUDA sebagai lapisan perantara untuk menangani permintaan alokasi dan pembebasan GPU secara efisien. Mekanisme ini memastikan manajemen sumber daya GPU yang efektif, yang berkontribusi pada kecepatan dan responsivitas keseluruhan aplikasi PyTorch[10].

PyTorch semakin menjadi alat penting bagi praktisi deep learning. Kumpulan fungsionalitas tingkat rendah yang luas memberikan kontrol mendetail atas jaringan neural dari pelatihan hingga inferensi. Tingkat kontrol ini sangat bermanfaat dalam penelitian deep learning, di mana studi ablasi yang mendalam dilakukan untuk memvalidasi arsitektur neural baru[11].

Salah satu alasan utama kesuksesan PyTorch adalah desainnya yang Pythonic, menawarkan antarmuka yang familiar dan intuitif bagi programmer Python[11]. Selain itu, PyTorch sangat dihargai karena fungsionalitas intinya yang ditulis dalam C++ untuk memastikan kinerja tinggi. Kombinasi antara kemudahan penggunaan ala Python dan optimisasi kinerja C++ menjadikan PyTorch alat yang kuat dan efisien untuk lingkungan penelitian dan produksi [11].

Sebagai kesimpulan, arsitektur PyTorch yang dinamis dan fleksibel, dikombinasikan dengan teknik optimisasi kinerja canggih dan antarmuka Pythonic, menjadikannya framework yang vital di bidang deep learning. Kemampuannya untuk menangani model kompleks dengan efisien dan memberikan kontrol mendetail kepada peneliti menegaskan pentingnya dalam kemajuan teknologi machine learning yang sedang berlangsung.

## *FACE DETECTION*



Gambar 2. 2 Gambar ilustrasi deteksi wajah (Sumber : https://studentsforhumanity.it)

Saat ini, teknologi pengenalan objek menggunakan kamera adalah salah satu topik paling menantang, di mana akurasi dan performa *real-time* telah menjadi indikator utama yang digunakan dalam sensor kamera[5]. *Face detection* merupakan salah satu masalah paling populer dalam *computer vision* yang melibatkan identifikasi wajah dalam gambar digital. Baru-baru ini, teknik *face detection* telah berkembang dari metode *computer vision* tradisional ke pendekatan *machine learning* (ML) yang lebih canggih. Tahapan utama dalam teknologi *face detection* meliputi menemukan area dalam gambar di mana terdapat wajah atau beberapa wajah[12].

Face detection, merupakan sebuah tugas yang dilakukan manusia dengan mudah, namun menimbulkan tantangan signifikan bagi mesin[12]. Sebagai masalah dasar dalam computer vision, deteksi wajah melibatkan penemuan wajah dalam gambar dan berfungsi sebagai langkah awal untuk banyak teknologi terkait wajah. Teknologi-teknologi ini termasuk verifikasi wajah, pemodelan wajah, pelacakan pose kepala, pengenalan jenis kelamin dan usia, pengenalan ekspresi wajah, dan banyak lagi[12].

*Face detection* adalah masalah *computer vision* yang melibatkan pencarian wajah dalam gambar. Ini juga merupakan langkah awal untuk banyak teknologi terkait wajah, seperti verifikasi wajah, pemodelan wajah, pelacakan pose kepala, pengenalan jenis kelamin dan usia, pengenalan ekspresi wajah[12] yang dapat membantu dalam melakukan deteksi kantuk pada pengemudi.

Kesimpulannya, meskipun deteksi wajah mungkin merupakan tugas sepele bagi manusia, ini mewakili tantangan yang kompleks dan menarik bagi mesin. Kemajuan yang terus-menerus di bidang ini membuka jalan bagi teknologi terkait wajah yang lebih canggih dan andal, memberikan kontribusi signifikan pada berbagai aplikasi di berbagai industri. Seiring dengan kemajuan penelitian, kita dapat mengharapkan sistem deteksi wajah yang lebih efisien dan akurat muncul, semakin memperkecil kesenjangan antara kemampuan persepsi visual manusia dan mesin.

## *APPLE CORE ML*

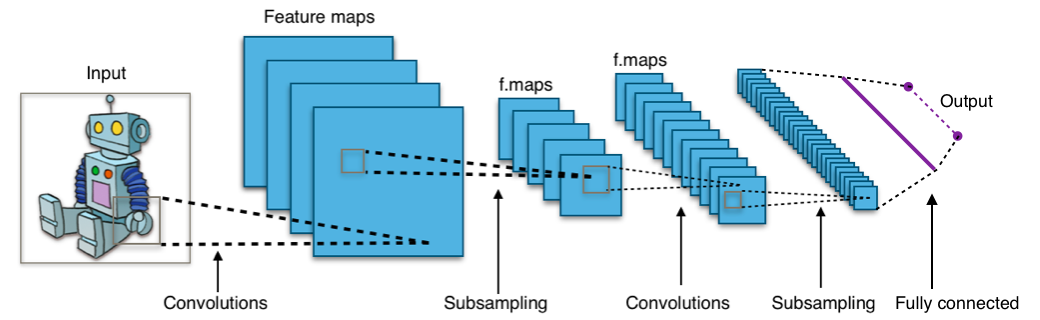


Gambar 2. 3 Gambar ilustrasi Apple core ML (Sumber : https://developer.apple.com/machine-learning/core-ml/)

*Apple Core* ML, pertama kali diperkenalkan pada tahun 2017, merupakan *framework* *machine learning* yang memungkinkan pengembang untuk mengintegrasikan model *machine* *learning* ke dalam aplikasi mereka dengan lebih mudah dan efisien. *Core* ML dirancang untuk memanfaatkan perangkat keras dari perangkat *Apple*, termasuk CPU, GPU, dan *Neural Engine*, guna memaksimalkan kinerja sambil meminimalkan penggunaan daya dan memori. Dengan *Core* ML, pengembang dapat membuat, melatih, dan mengonversi model *machine learning* dari berbagai pustaka pihak ketiga menggunakan paket *Python coremltools*, serta memanfaatkan *Xcode* untuk mengamankan dan mengoptimalkan model mereka.

Selain itu, *Core* ML menyediakan berbagai alat dan fitur untuk meningkatkan pengalaman pengembang dan pengguna. Alat-alat ini termasuk dukungan untuk model enkripsi, profil aplikasi untuk memantau panggilan API *Core* ML, serta *preview* langsung dari perilaku model pada data sampel atau dari kamera dan mikrofon perangkat. *Apple* juga menyediakan *Create* ML, alat yang memungkinkan pengembang untuk membangun dan melatih model *Core* ML langsung di *Mac* mereka tanpa perlu menulis kode. Dengan dukungan ekstensif dan dokumentasi yang lengkap, *Core* ML memungkinkan pengembang untuk dengan mudah mengintegrasikan teknologi *machine learning* yang canggih ke dalam aplikasi mereka, membuat aplikasi lebih pintar dan responsif terhadap kebutuhan pengguna​.

## Convolutional Neural Networks



Gambar 2. 4 Gambar ilustrasi algoritma CNN (Sumber : https://www.researchgate.net)

Selama sepuluh tahun terakhir, *Convolutional Neural Networks* (CNN) telah menjadi standar utama untuk berbagai operasi dalam bidang *computer vision* dan *machine learning*[13]. *Convolutional Neural Networks* (CNN) merupakan kelas dari jaringan saraf *deep feed-forward*, yang dibangun dengan meniru pola koneksi *neuron* di korteks visual manusia[14].

*Convolutional Neural Networks* dapat secara otomatis mempelajari fitur-fitur yang mewakili variasi visual kompleks dari data pelatihan yang banyak.[15]. Salah satu keberhasilan *Convolutional Neural Networks* adalah dalam pengenalan wajah. Pengenalan wajah oleh komputer adalah teknik yang memungkinkan komputer secara otomatis mengenali wajah dalam gambar. Berbagai peneliti telah melakukan penelitian terkait *face detection*[15].

Convolutional Neural Networks (CNNs) telah membuat kemajuan yang luar biasa, terutama dalam pengenalan wajah, sebuah tugas penting dalam visi komputer. Teknologi ini memungkinkan komputer untuk secara otomatis mengidentifikasi wajah dalam gambar, menandai tonggak penting dalam penelitian kecerdasan buatan. Peneliti di seluruh dunia telah mendedikasikan usaha yang cukup besar untuk memajukan algoritma pengenalan wajah, berkontribusi pada penyempurnaan dan efektivitas CNN dalam bidang ini[15].

Jaringan Neural Konvolusional terdiri dari beberapa lapisan. Lapisan-lapisan tersebut disusun secara berurutan yaitu :

### Convolutional Layer

Di inti arsitektur CNN terletak Lapisan Konvolusi, bertindak sebagai elemen dasar dari jaringan. Lapisan ini terdiri dari beberapa filter, atau kernel, masing-masing berisi nilai-nilai numerik. Filter-filter ini memainkan peran penting dalam menghasilkan peta fitur, mengekstraksi fitur visual penting dari lapisan masukan atau peta fitur lapisan sebelumnya [15].

### Pooling Layer

Setelah Lapisan Konvolusi adalah Lapisan Pooling, ditempatkan secara strategis untuk mengurangi dimensi dari peta fitur yang dihasilkan oleh lapisan konvolusi. Dengan menggunakan teknik seperti max-pooling atau average pooling, lapisan ini secara efektif menyederhanakan informasi dari lapisan sebelumnya, mengoptimalkan efisiensi komputasi sambil mempertahankan fitur-fitur penting[15].

### Fully Connected Layer

Selanjutnya, Lapisan Koneksi Penuh bertindak sebagai pusat di mana setiap node terhubung dengan semua node dari peta fitur yang berasal dari lapisan pooling terakhir. Lapisan ini menggabungkan fitur yang diekstraksi dan memprosesnya untuk menghasilkan output akhir. Jumlah node output dalam lapisan ini sesuai dengan tugas spesifik yang sedang dihadapi, seperti jumlah kelas objek yang akan diidentifikasi dalam tugas pengenalan objek[15].

### Activation Function

Faktor penting yang memengaruhi kinerja CNN adalah fungsi aktivasi, yang memperkenalkan non-linearitas ke dalam jaringan dan memfasilitasi pengenalan pola yang kompleks. Berbagai fungsi aktivasi, termasuk sigmoid, tanh, Rectified Linear Units (ReLU), antara lain, digunakan, dengan ReLU menjadi yang paling umum karena kesederhanaan dan efektivitasnya[15].

CNN menunjukkan kemampuan yang mengesankan untuk secara otomatis belajar representasi visual yang rumit dari dataset yang luas, memungkinkannya untuk memahami variasi visual yang kompleks dengan akurasi yang luar biasa[15]. Kemampuan bawaannya ini membuat CNN menjadi alat yang tangguh dalam berbagai tugas pengenalan gambar, menyoroti pentingnya dalam bidang kecerdasan buatan[10].

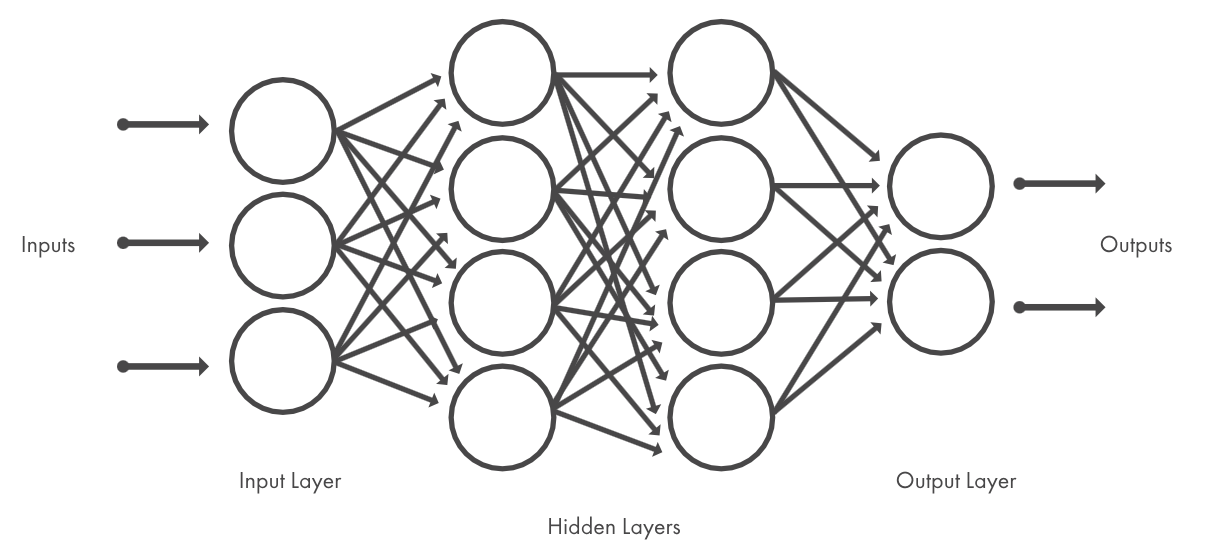
Selain itu, CNN mewakili mekanisme pemetaan yang kuat antara data masukan dan keluaran, mampu mempelajari hubungan pemetaan yang luas melalui proses pelatihan yang cermat. Beroperasi di bawah kerangka pembelajaran mendalam yang disupervisi, CNN terus memperbaiki pemahamannya tentang pola data yang kompleks .

Keuntungan bawaan dari CNN terletak pada ukuran masukan yang relatif kecil dari setiap neuron, memfasilitasi propagasi gradien yang efektif melintasi banyak lapisan jaringan. Hal ini memperkuat kedalaman jaringan dan efektivitasnya dalam mengidentifikasi pola yang rumit, meningkatkan kinerja secara keseluruhan.

Di bidang pengenalan wajah, teknologi berbasis CNN telah muncul sebagai pendekatan dominan, merevolusi bidang tersebut dengan akurasi dan efisiensi yang belum pernah terjadi sebelumnya. Memanfaatkan prinsip yang terinspirasi dari pola koneksi neuron dalam korteks visual manusia, CNN mewakili perubahan paradigma dalam cara mesin memahami dan menafsirkan informasi visual[10].

Secara kesimpulan, Convolutional Neural Networks berdiri sebagai salah satu algoritma utama dalam pembelajaran mendalam, mendorong kemajuan dalam pengenalan gambar, khususnya dalam pengenalan wajah. Arsitektur multi-lapisannya, ditambah dengan kemampuannya yang bawaan untuk mempelajari fitur-fitur kompleks dari data, menempatkan CNN di garis depan penelitian dan aplikasi kecerdasan buatan modern.

## DEEP LEARNING



Gambar 2. 5 Gambar ilustrasi tahapan pada deep learning (Sumber : https://www.researchgate.net)

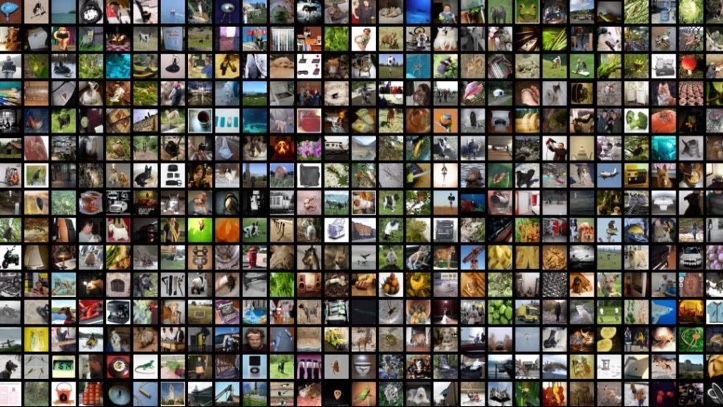
Deep Learning (DL), sebuah cabang penting dari machine learning (ML), telah muncul sebagai kekuatan yang kuat dalam kecerdasan buatan (AI). Berakar dalam pengolahan gambar dan pengenalan pola, jangkauan DL melampaui metode ML tradisional. Aplikasinya dalam pengenalan wajah, misalnya, menunjukkan keahliannya dalam menjelajahi kompleksitas wajah dengan presisi (1). Terdiri dari serangkaian algoritma yang canggih, DL unggul dalam mengurai pola kompleks dan menemukan wawasan yang berarti dari data. Beberapa tahun terakhir telah menyaksikan metode deep learning mendapatkan popularitas karena kemampuannya untuk belajar fitur secara otomatis dari data mentah (2). Deteksi objek, sebuah tantangan yang abadi, telah mengalami perubahan paradigma dengan munculnya deep learning, menarik perhatian peneliti secara global. Dianggap sebagai garda terdepan AI dan visi komputer, metode deep learning mendorong inovasi dalam klasifikasi gambar, deteksi objek, dan Pemrosesan Bahasa Alami (NLP) (3). Dampaknya dalam ranah aplikasi Unmanned Aerial Vehicle (UAV) khususnya mencolok, merevolusi pengenalan objek dengan akurasi dan efisiensi yang tak tertandingi.

*Deep Learning* (DL) adalah pencapaian terbaru dalam era *machine learning*, di mana awalnya mampu menunjukkan kemampuan mendekati manusia, dan kini kemampuan yang melampaui manusia dalam banyak aplikasi termasuk terjemahan suara-ke-teks, deteksi dan pengenalan objek, deteksi anomali, pengenalan emosi dari rekaman audio atau video, dan sebagainya[13].

Konsep paling penting dalam bidang *deep learning* terinspirasi oleh sistem biologis manusia yang cenderung fokus pada bagian-bagian khas saat memproses sejumlah informasi yang besar. Dengan perkembangan *deep neural networks*, perhatian mengenai mekanisme ini telah banyak digunakan dalam berbagai domain aplikasi yang sangat luas[16]. *Deep learning* adalah istilah yang digunakan untuk menyebut seperangkat metode dalam membangun jaringan saraf berlapis-lapis (atau *'deep'*) yang mampu menyelesaikan masalah-masalah menantang dalam *supervised classification*, pemodelan generatif, atau pembelajaran mendalam[17].

Model *deep learning* memiliki banyak potensi pada ilmu jaringan (*neuroscience*). Salah satu yang paling menarik adalah kemampuannya melampaui perancangan fungsi secara manual dan memahami bagaimana komputasi berkembang dari pengalaman. *Neuroscientists* sudah menyadari peluang ini, namun pemanfaatannya baru saja dimulai[17].

## *DATASET*



Gambar 2. 6 Gambar ilustrasi dataset gambar (Sumber : https://neurohive.io)

*Dataset* adalah kumpulan data yang terorganisir dan biasanya disimpan dalam bentuk tabel atau *database*, yang digunakan untuk analisis, pemrosesan, dan penelitian. Setiap *dataset* terdiri dari elemen-elemen yang dikenal sebagai '*data points*' atau '*records*', yang masing-masing memiliki sejumlah atribut atau variabel yang menjelaskan karakteristik data tersebut. Sebagai contoh, dalam *dataset* kesehatan, setiap rekaman mungkin berisi informasi tentang pasien seperti usia, jenis kelamin, diagnosis, dan hasil tes medis. *Dataset* dapat dikumpulkan dari berbagai sumber seperti eksperimen ilmiah, survei, sensor, transaksi bisnis, dan media sosial, dan sering kali dilengkapi dengan *metadata* yang menjelaskan struktur, asal, dan konteks data tersebut. Pengelolaan dan analisis *dataset* memainkan peran kunci dalam berbagai disiplin ilmu, termasuk ilmu data, *machine learning*, statistik, dan informatika, membantu peneliti dan profesional dalam membuat keputusan yang didukung oleh data dan mengembangkan wawasan baru.

*Dataset* telah muncul dalam ratusan makalah penelitian, di mana *dataset* tersebut digunakan sebagai dasar untuk pengembangan dan perbandingan banyak algoritma[18], *Datasets* sangat penting dalam ekosistem *machine learning*. Selain menyediakan data pelatihan dan pengujian bagi pembuat model, *datasets* juga merumuskan masalah, mengorganisir komunitas, dan menjembatani antara akademia dan industri[18].

Berikut adalah daftar peran dataset pada penelitian ini :

1. Sumber Data Utama:

*Dataset* yang Anda miliki, yang terdiri dari foto-foto pengemudi saat mengantuk dan sadar, merupakan sumber data utama yang digunakan untuk melatih dan menguji algoritma YOLOv8. Tanpa dataset ini, model deteksi kantuk tidak dapat dibuat atau diuji.

1. Pelatihan Model:

*Dataset* digunakan untuk melatih model YOLOv8 agar dapat mengenali dan mendeteksi tanda-tanda kantuk pada pengemudi. Dalam proses pelatihan, model akan mempelajari pola-pola visual yang terkait dengan kondisi mengantuk dan sadar berdasarkan gambar-gambar yang ada dalam *dataset*.

1. Validasi dan Pengujian:

Setelah model dilatih, *dataset* yang berbeda digunakan untuk validasi dan pengujian. Hal ini penting untuk mengevaluasi kinerja model dan memastikan bahwa model dapat mendeteksi kantuk dengan akurasi yang tinggi pada gambar-gambar yang belum pernah dilihat sebelumnya.

1. *Preprocessing* dan *Augmentasi*:

*Dataset* juga memainkan peran dalam tahap *preprocessing*, di mana gambar-gambar mungkin perlu diubah ukurannya, dinormalisasi, atau dilakukan *augmentasi* (seperti rotasi, *flipping*, atau perubahan kecerahan) untuk meningkatkan kualitas dan variasi data yang tersedia untuk pelatihan model.

1. Pemetaan Kelas:

*Dataset* harus diberi label dengan benar untuk memastikan bahwa setiap gambar dikategorikan dengan tepat sebagai "mengantuk" atau "sadar." Label ini sangat penting agar model dapat belajar membedakan antara kedua kondisi tersebut.

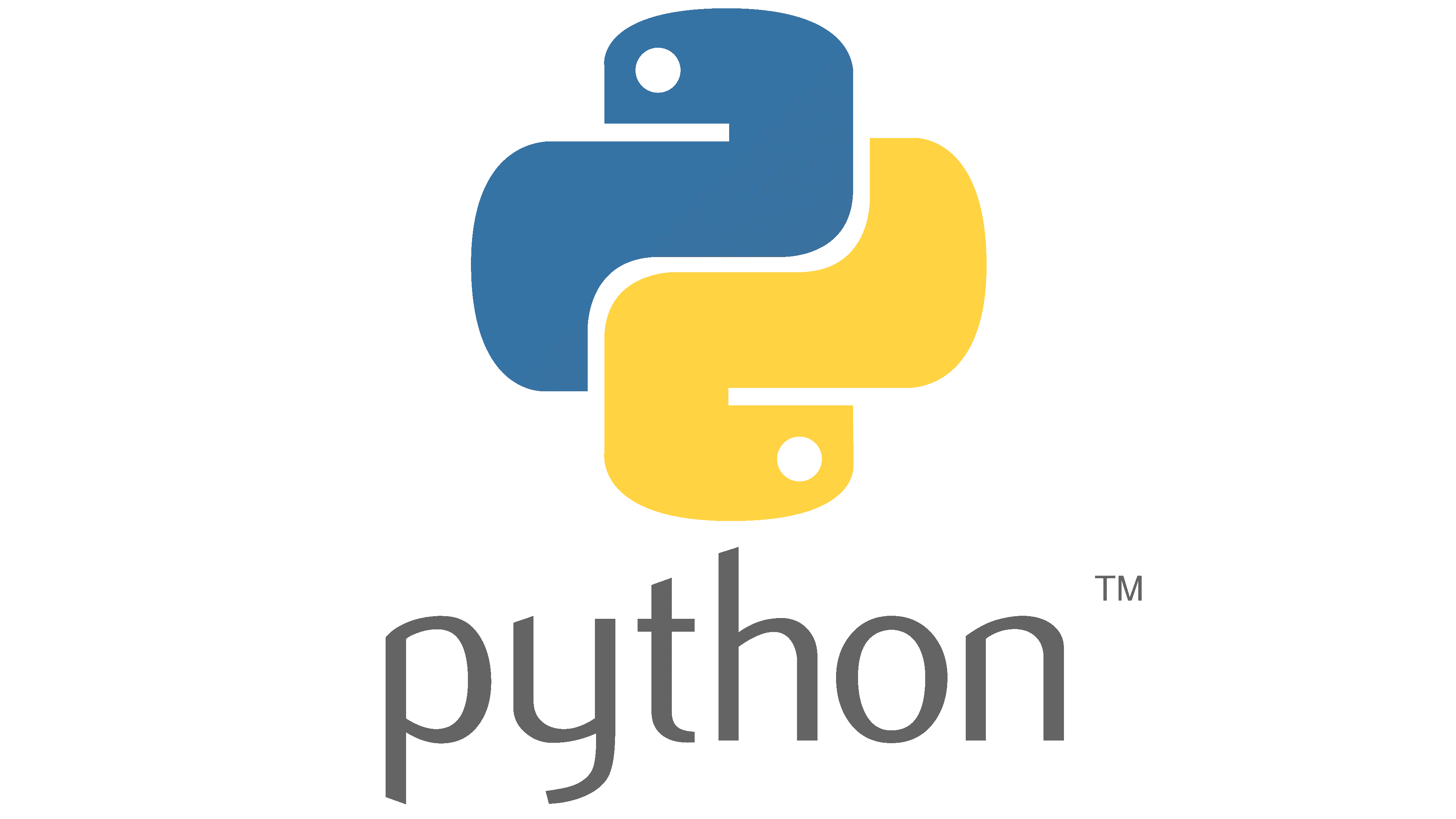
1. Evaluasi Model:

Kualitas dan keberagaman *dataset* mempengaruhi evaluasi model. *Dataset* yang baik harus mencakup berbagai kondisi pencahayaan, sudut pandang, dan variasi wajah untuk memastikan model yang dihasilkan dapat diandalkan dalam situasi dunia nyata.

1. Reproduksibilitas Penelitian:

Dengan mendokumentasikan dan mungkin membagikan *dataset* (dengan memperhatikan privasi dan etika), penelitian Anda dapat direproduksi oleh peneliti lain, yang meningkatkan validitas dan kontribusi ilmiah dari skripsi Anda.

## *Python*



Gambar 2. 7 Gambar Logo Python (Sumber : https://www.ntuclearninghub.com)

Pada tahun 1991, Guido van Rossum menciptakan bahasa pemrograman *Python*. Ternyata, ada cerita menarik di balik penamaan "*Python*". Saat itu, Guido sedang membaca naskah seri BBC "Monty Python's Flying Circus". Terinspirasi oleh buku tersebut, ia memutuskan untuk memberi nama bahasa pemrogramannya "*Python*" karena terdengar singkat dan unik. *Python* adalah bahasa pemrograman yang berorientasi objek, diinterpretasikan, dan interaktif. Bahasa ini menawarkan berbagai struktur data tingkat tinggi seperti *list*, *tuple*, *set*, dan *array* asosiatif (disebut *dictionary*). *Python* juga mendukung pengetikan dan pengikatan dinamis, serta menyediakan modul, kelas, pengecualian, dan manajemen memori otomatis[19].

Selain itu, *Python* digunakan dalam sistem komputasi paralel dan terkenal dengan sintaksnya yang sederhana dan mudah dipahami, namun tetap kuat dan fleksibel. *Python* juga memiliki interpreter untuk Java yang dikenal sebagai *JPython*, mirip dengan interpreter untuk bahasa C. Salah satu keunggulan *Python* adalah ketersediaan berbagai pustaka yang luas, yang bisa mengurangi jumlah kode yang harus ditulis oleh *programmer* hingga sepertiga. Inilah salah satu alasan mengapa *Python* telah mencapai puncak popularitas dalam bidang Pembelajaran Mesin (*Machine Learning*)[19].

## Penelitian Terdahulu

Tabel 2. 2 Tabel penelitian terdahulu

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Judul | Penulis | Metode | Hasil | Pembaruan | Keterkaitan |
|  |  |  |  |  |  |  |
| 1 | *IoT-Based Non-Intrusive Automated Driver Drowsiness Monitoring Framework for Logistics and Public Transport Applications to Enhance Road Safety* | M. ADIL KHAN, TAHIR NAWAZ , UMAR S. KHAN , (Member, IEEE), AMIR HAMZA, AND NASIR RASHID | Pelatihan algoritma Haar Cascade untuk pengenalan wajah. Algoritma ini kemudian diterapkan pada aplikasi berbasis Android untuk mendeteksi wajah pengemudi secara otomatis. Sistem ini bertujuan untuk memantau tanda-tanda kelelahan pada pengemudi dalam konteks logistik dan transportasi umum, sehingga dapat meningkatkan keselamatan jalan raya. | Jurnal ini mengusulkan kerangka kerja IoT otomatis *end-to-end* untuk memantau perilaku pengemudi, dirancang khusus untuk aplikasi logistik dan transportasi publik. Sistem ini terdiri dari Raspberry Pi dan Nvidia Jetson Nano untuk *edge computing* yang terhubung dengan *webcam* untuk menangkap gambar secara *real-time, cloud computing*, dan aplikasi *mobile*. Deteksi kantuk dilakukan dengan menganalisis aktivitas mata dan mulut, seperti tidur dan menguap, menggunakan teknik pemrosesan gambar berbasis *Deep Neural Network* (DNN) dengan *multilayer perceptron classifiers*. Sistem ini memantau empat parameter perilaku pengemudi: aktif, menguap, mata tertutup, dan terdistraksi, dengan akurasi deteksi sebesar 96%. Data yang dikumpulkan dikirim ke *database real-time* di *cloud*, memungkinkan pemantauan jarak jauh melalui aplikasi Android interaktif, di mana admin dapat menambahkan beberapa pengemudi dan menerima notifikasi kantuk serta informasi terkait lainnya untuk evaluasi pengemudi. | Pembaruan dalam penelitian ini meliputi penerapan sistem menggunakan aplikasi Android yang terintegrasi dengan server, memungkinkan pemantauan kondisi pengemudi secara real-time. Integrasi ini bertujuan untuk meningkatkan efektivitas deteksi kelelahan pengemudi pada aplikasi berbasis Android, mendukung pengawasan dan peningkatan keselamatan dalam lalu lantas | Fokus pada deteksi kantuk pengemudi menggunakan IoT dan otomatisasi. |
| 2 | *Drowsiness Detection Based On Driver Temporal Behavior Using a New Developed Dataset* | F. Faraji, F. Lotfi, J. Khorramdel, A. Najafi, A. Ghaffari K. N. Toosi University of Technology, Tehran, Iran. | Penggunaan algoritma YOLOv3 untuk deteksi objek. Algoritma ini kemudian digabungkan dengan LSTM (Long Short-Term Memory) dalam proses pelatihan data untuk menganalisis perilaku temporal pengemudi. Kombinasi ini bertujuan untuk meningkatkan akurasi deteksi tanda-tanda kelelahan pada pengemudi. | Studi ini menggunakan pendekatan berbasis gambar dengan akurasi yang memadai dan proses yang cepat. Kami menerapkan YOLOv3 (*You Look Only Once-version*3) CNN (*Convolutional Neural Network*) untuk mengekstrak fitur wajah secara otomatis. Selanjutnya, jaringan neural LSTM (*Long-Short Term Memory*) digunakan untuk mempelajari perilaku temporal pengemudi termasuk periode waktu mengantuk dan kedipan mata serta klasifikasi urutan. Untuk melatih YOLOv3, kami menggunakan *dataset* yang dikumpulkan bersama metode *transfer learning*. Selain itu, dataset untuk proses pelatihan LSTM dihasilkan oleh CNN yang disebutkan dan diformat sebagai urutan dua dimensi yang terdiri dari durasi waktu kedipan mata dan mengantuk. *Dataset* yang dikembangkan mempertimbangkan gangguan seperti pencahayaan dan postur kepala pengemudi. Untuk eksperimen *real-time*, sebuah kerangka *multi thread* dikembangkan untuk menjalankan kedua CNN dan LSTM secara paralel. Hasilnya menunjukkan kemampuan hibrida CNN dan LSTM dalam mendeteksi kantuk serta efektivitas metode yang diusulkan. | Peningkatan efektivitas sistem melalui integrasi algoritma YOLOv3 dengan LSTM. LSTM digunakan untuk menganalisis data urutan waktu (temporal data) dari perilaku pengemudi, memungkinkan sistem untuk mendeteksi tanda-tanda kelelahan dengan lebih akurat berdasarkan pola perilaku yang berubah seiring waktu. | Menggunakan yolo sebagai algoritma utama untuk melakukan pengembangan teknologi deteksi wajah mengantuk/lelah bagi pengemudi |
| 3 | *Vision Transformers and YoloV5 based Driver Drowsiness Detection Framework* | Ghanta Sai Krishna, Kundrapu Supriya, Jai Vardhan and Mallikharjuna Rao K | Penerapan YOLOv5 untuk deteksi objek. Algoritma YOLOv5 kemudian digabungkan dengan Vision Transformers untuk menganalisis dan mengklasifikasikan fitur visual dari wajah dan perilaku pengemudi. Kombinasi ini bertujuan untuk meningkatkan akurasi deteksi tanda-tanda kelelahan pada pengemudi | Metode yang digunakan dalam penelitian ini adalah penggunaan *framework* berbasis *vision transformers* dan arsitektur YOLOv5 untuk pengenalan kantuk pengemudi. Mereka mengusulkan arsitektur khusus *pre*-*trained* YOLOv5 untuk ekstraksi wajah dengan tujuan mengekstraksi *Region of Interest* (ROI). Selain itu, karena keterbatasan arsitektur sebelumnya, mereka memperkenalkan *vision transformers* untuk klasifikasi gambar biner yang dilatih dan divalidasi pada *dataset* publik UTA-RLDD. Model ini mencapai akurasi pelatihan sebesar 96,2% dan akurasi validasi sebesar 97,4%. Selama pelatihan, teknik *augmentasi* gambar digunakan untuk menghasilkan variasi gambar yang dapat meningkatkan kinerja *framework* ViT. Dengan demikian, metode yang dicapai adalah akurasi validasi sebesar 97.4%. | Penerapan YOLOv5 yang dibantu dengan Vision Transformers untuk meningkatkan akurasi klasifikasi. Integrasi Vision Transformers membantu sistem dalam memahami dan menganalisis fitur visual secara lebih mendalam, sehingga meningkatkan kemampuan deteksi kantuk pada pengemudi. | Penerapan YOLO dalam mengembangkan teknologi deteksi wajah lelah pengemudi. |
| 4 | Aplikasi *Warning Alert* Pendeteksi Kelelahan Ekspresi Wajah Pada Pengemudi Secara *Real-Time* Menggunakan Metode *You Only Look Once* Berbasis *Website* | Hafidh Ahmad Fauzan , Ari Kurniawan | Pelatihan yolov4 dengan dataset RAKIBUL.ECE.RUET. Algoritma yolov4 diterapkan untuk mendeteksi ekspresi wajah yang menunjukkan tanda-tanda kelelahan pada pengemudi secara real-time melalui aplikasi berbasis web. Kombinasi ini bertujuan untuk meningkatkan akurasi dan efisiensi deteksi kelelahan pada pengemudi. | Penelitian yang menggunakan yolov4 untuk mendeteksi kelelahan pada pengemudi berdasarkan ekspresi wajah secara *real-time*. Algoritma pendeteksian diterapkan melalui platform *website* yang terhubung dengan Telegram, memungkinkan pengiriman pesan secara langsung mengenai status dan kondisi pengemudi dengan detail. Penulis memanfaatkan *dataset* yang tersedia di Kaggle, dipublikasikan oleh RAKIBUL.ECE.RUET, yang terdiri dari 4560 *file* per kelas, yaitu *active* dan *fatigue*. Selain itu, penulis juga menyiapkan *dataset* tambahan yang telah di-*augmentasi* menggunakan *photometric distortion* dan *geometric distortion* untuk meningkatkan variasi data. Melalui pendekatan ini, penulis berhasil mengembangkan metode baru yang mencapai akurasi sebesar 97,03%, menunjukkan efektivitas sistem dalam mendeteksi kelelahan pengemudi secara akurat dan cepat. | Penggunaan website serta bot Telegram untuk membantu mengirimkan informasi kepada pengemudi ataupun pengawas. Integrasi ini memungkinkan pemantauan dan pemberian peringatan secara real-time, sehingga meningkatkan respons terhadap tanda-tanda kelelahan pada pengemudi. | Implementasi YOLO untuk teknologi deteksi kelelahan ekspresi wajah pengemudi. |
| 5 | Implementasi Algoritma *Deep Learning* Untuk Sistem Deteksi Kantuk Pada Pengemudi MenggunakanYOLO | Mamta Anisa Bella | Pelatihan algoritma yolov4 menggunakan dataset open source. Algoritma yolov4 diterapkan untuk mendeteksi ekspresi wajah yang menunjukkan tanda-tanda kantuk pada pengemudi. Penelitian ini fokus pada pengujian berbagai variasi pelatihan untuk meningkatkan akurasi deteksi kantuk secara keseluruhan. | Dalam penelitian ini, pengemudi yang mengantuk diidentifikasi dengan label "*drowsy*", sementara pengemudi yang terjaga diberi label "*awake*". Metode yang digunakan memanfaatkan kombinasi yolov4 dan CNN, diimplementasikan melalui sebuah *website* yang terhubung langsung dengan layanan Telegram untuk mengirimkan status pengemudi secara *real-time*. *Dataset* yang digunakan diambil dari Kaggle, yang telah dipublikasikan oleh RAKIBUL.ECE.RUET, dengan masing-masing kelas memiliki 4560 *file*, serta *dataset* tambahan yang telah melalui proses *augmentasi*. Konfigurasi model termasuk penggunaan *batch* *size* 64, ukuran jaringan 416x416, subdivisi 16, *max* *batch* 4000, dan filter 21. Pembagian *dataset* 80%:20% dan *learning rate* 0.00261 memberikan nilai *Intersection* *over Union* (iou) terbesar. Dengan metode ini, sistem berhasil mencapai akurasi sebesar 97,03%, menunjukkan efektivitasnya dalam mendeteksi kelelahan pengemudi dengan akurasi dan kecepatan yang tinggi. | Melakukan uji coba variasi pelatihan untuk memperoleh akurasi maksimal. Dengan menguji berbagai konfigurasi dan parameter pelatihan, penelitian ini bertujuan untuk mengoptimalkan kinerja algoritma yolov4 dalam mendeteksi tanda-tanda kantuk pada pengemudi. | Pemanfaatan YOLO sebagai algoritma utama untuk pengembangan teknologi deteksi kelelahan wajah pengemudi. |
| 6 | *Driver's Drowsiness Detection System* | Aayush Bhetuwal, Siddanta K C | Pelatihan yolov5 agar dapat melakukan deteksi wajah. Algoritma yolov5 diterapkan untuk mendeteksi ekspresi wajah yang menunjukkan tanda-tanda kantuk pada pengemudi. Penelitian ini mengoptimalkan proses pelatihan dengan menggunakan CUDA dan Adam optimizer untuk meningkatkan efisiensi dan akurasi deteksi. | Penelitian ini mempersembahkan sistem deteksi kantuk pengemudi yang inovatif dengan memanfaatkan kemampuan canggih YOLOv5 dan CNN. Menggunakan *dataset* yang terdiri dari 4560 gambar aktif dan 4560 gambar kelelahan yang dikumpulkan dari Kaggle, sistem ini menunjukkan kinerja yang luar biasa. Model YOLOv5s mencapai akurasi mengesankan sebesar 93,01% dan *Intersection over Union* (IoU) sebesar 95,51%, meskipun hanya dilatih selama 10 *epoch*. Kemampuan deteksi *real-time* dari sistem ini, dikombinasikan dengan tingkat akurasinya yang tinggi, menunjukkan potensi signifikan untuk meningkatkan keselamatan di jalan raya dengan memantau dan merespons kelelahan pengemudi secara efektif. | Optimalisasi pelatihan menggunakan CUDA dan Adam optimizer untuk memaksimalkan efisiensi pelatihan. Penggunaan CUDA memungkinkan pemrosesan paralel pada GPU, sementara Adam optimizer meningkatkan kecepatan konvergensi dan stabilitas model selama pelatihan. | Kedua penelitian ini memiliki fokus pada deteksi kelelahan pada pengemudi menggunakan teknologi deteksi visual, meskipun menggunakan versi YOLO yang berbeda. |

# Metodologi Penelitian

## Metodologi Penelitian

Dalam penelitian ini, tujuan utama adalah mengembangkan dan menguji model deteksi kantuk pada pengemudi menggunakan algoritma YOLOv8. Mengingat pentingnya keselamatan berkendara dan potensi bahaya yang ditimbulkan oleh pengemudi yang mengantuk, deteksi kantuk secara otomatis menjadi topik yang krusial. Untuk mencapai tujuan ini, kami mengadopsi pendekatan berbasis data yang memungkinkan pengembangan model yang akurat dan handal. Penelitian ini berfokus pada pengumpulan data nyata dari simulasi mengemudi, di mana data ini digunakan untuk melatih dan menguji algoritma YOLOv8. Dengan pendekatan ini, kami dapat mengevaluasi efektivitas algoritma dalam mendeteksi tanda-tanda kantuk pada pengemudi dalam kondisi yang dikontrol secara ketat. Oleh karena itu, metode penelitian yang digunakan dalam studi ini adalah metode kuantitatif, yang lebih tepatnya adalah eksperimen.

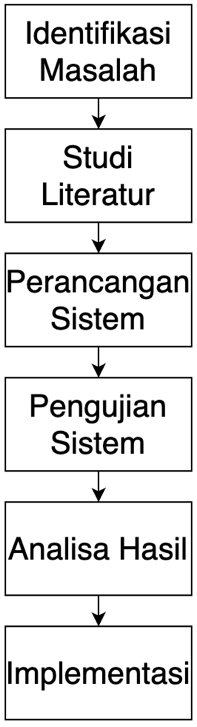
### Metode Kuantitatif

Dalam penelitian ini, tujuan utama adalah mengembangkan dan menguji model deteksi kantuk pada pengemudi menggunakan algoritma YOLOv8. Mengingat pentingnya keselamatan berkendara dan potensi bahaya yang ditimbulkan oleh pengemudi yang mengantuk, deteksi kantuk secara otomatis menjadi topik yang krusial. Untuk mencapai tujuan ini, kami mengadopsi pendekatan berbasis data yang memungkinkan pengembangan model yang akurat dan handal. Penelitian ini berfokus pada pengumpulan data nyata dari simulasi mengemudi, di mana data ini digunakan untuk melatih dan menguji algoritma YOLOv8. Dengan pendekatan ini, kami dapat mengevaluasi efektivitas algoritma dalam mendeteksi tanda-tanda kantuk pada pengemudi dalam kondisi yang dikontrol secara ketat.

Untuk mencapai tujuan penelitian ini, kami menggunakan metode kuantitatif dengan pendekatan eksperimen. Kami mengumpulkan data dengan memotret diri sendiri dalam berbagai kondisi "awake" dan "drowsy". Proses ini dilakukan dengan mengambil banyak variasi foto yang mencerminkan kondisi terjaga dan mengantuk. Foto-foto ini kemudian dilabeli secara manual untuk menandai tanda-tanda kantuk seperti mata tertutup, menguap, dan kepala yang menunduk. Data set yang kami kumpulkan ini digunakan untuk melatih dan menguji model YOLOv8. Selama pelatihan, model dilatih untuk mengenali dan mendeteksi tanda-tanda kantuk berdasarkan data yang telah dilabeli. Hasil eksperimen menunjukkan bahwa model YOLOv8 mampu mendeteksi tanda-tanda kantuk dengan tingkat akurasi yang tinggi, menunjukkan potensi besar dalam meningkatkan keselamatan berkendara.

## Kerangka Penelitian

Kerangka berpikir atau kerangka pemikiran merupakan dasar pemikiran dari penelitian yang disintesiskan dari fakta-fakta, observasi, dan kajian kepustakaan. Kerangka ini berfungsi sebagai panduan bagi peneliti dalam menganalisis dan merencanakan penelitian, serta membentuk argumen yang mendasari asumsi dan hipotesis yang diajukan[20]. Dengan mengintegrasikan berbagai sumber informasi, peneliti dapat mengidentifikasi pola, hubungan, dan kecenderungan yang relevan dengan topik penelitian. Kerangka berpikir tidak hanya membantu dalam merumuskan tujuan dan metodologi penelitian, tetapi juga dalam mengarahkan interpretasi data dan kesimpulan yang akan diambil. Dengan demikian, kerangka berpikir merupakan komponen esensial yang memastikan penelitian berjalan secara sistematis dan berbasis pada landasan ilmiah yang kuat. Kerangka penelitian yang kami lakukan dapat diuraikan sebagai berikut:



Gambar 3. 1 Gambar kerangka penelitian

### Identifikasi Masalah

1. Pengumpulan Dataset

Dalam penelitian ini, pengumpulan dataset dilakukan dengan mengambil foto secara mandiri sebanyak 200 foto dalam berbagai kondisi. Proses ini memerlukan waktu dan ketelitian tinggi karena kami harus memastikan bahwa data yang dikumpulkan mencakup berbagai variasi kondisi pencahayaan, sudut pandang kamera, dan ekspresi wajah pengemudi. Pengambilan foto dilakukan di berbagai situasi, baik di siang hari dengan pencahayaan terang maupun di malam hari dengan pencahayaan minim. Tantangan utama dalam tahap ini adalah memastikan bahwa setiap foto yang diambil memiliki kualitas yang cukup baik untuk digunakan dalam pelatihan model, serta mengatasi gangguan yang mungkin timbul selama proses pengambilan gambar.

1. Anotasi Gambar

Setelah dataset terkumpul, langkah berikutnya adalah anotasi gambar menggunakan alat *LabelImg*. Proses ini sangat memakan waktu dan membutuhkan ketelitian tinggi karena setiap foto harus dianotasi dengan akurat untuk menandai area wajah dan tanda-tanda kantuk. Mengingat jumlah data yang cukup banyak, proses anotasi menjadi sangat melelahkan dan memerlukan dedikasi penuh. Selain itu, menjaga konsistensi dalam anotasi sangat penting untuk memastikan kualitas dataset. Anotasi yang tidak konsisten dapat mengakibatkan model yang dilatih menjadi kurang akurat.

1. Pelatihan Model

Pelatihan model dilakukan menggunakan kartu grafis RTX 2080, yang memberikan kecepatan dan efisiensi dalam proses pelatihan. Meskipun demikian, proses pelatihan tetap merupakan tahap yang sangat teknis dan menantang. Kami harus mencoba berbagai konfigurasi hyperparameter untuk mendapatkan hasil yang optimal, serta mengatasi masalah overfitting dan underfitting yang sering muncul. Setiap iterasi pelatihan membutuhkan waktu yang cukup lama, sehingga proses ini memerlukan banyak kesabaran dan ketekunan. Penggunaan RTX 2080 sangat membantu dalam mempercepat proses pelatihan, tetapi tantangan teknis tetap ada dalam memastikan model bekerja dengan baik.

1. Pengujian Model

Setelah model dilatih, pengujian dilakukan untuk memastikan akurasi dan kinerjanya dalam mendeteksi tanda-tanda kantuk. Pengujian melibatkan perhitungan F1 score serta perbandingan akurasi menggunakan data yang belum pernah dilihat oleh model. Tantangan utama dalam tahap ini adalah memastikan bahwa model dapat mendeteksi kantuk dengan akurat dalam berbagai kondisi pencahayaan dan posisi kamera. Kami juga harus menguji model dalam berbagai skenario nyata untuk memastikan bahwa performanya tidak hanya baik dalam kondisi laboratorium tetapi juga dalam situasi kehidupan nyata yang lebih kompleks.

1. Konversi Model

Tahap terakhir adalah konversi model agar dapat diintegrasikan dengan Apple Core ML untuk digunakan pada perangkat iOS. Proses ini melibatkan penulisan kode khusus untuk memastikan bahwa model dapat berjalan dengan efisien di perangkat iOS tanpa mengurangi akurasinya. Tantangan dalam tahap ini termasuk mengatasi keterbatasan perangkat mobile dalam hal sumber daya komputasi serta memastikan bahwa model tetap efisien dan cepat. Menulis kode konversi sendiri memungkinkan kami untuk mengoptimalkan model sesuai dengan kebutuhan spesifik perangkat yang akan digunakan.

### Studi Literatur

Studi literatur merupakan langkah penting dalam penelitian yang bertujuan untuk memahami dan merangkum pengetahuan yang sudah ada terkait topik penelitian. Melalui studi literatur, peneliti dapat mengidentifikasi kesenjangan dalam penelitian sebelumnya, mengembangkan kerangka teoretis yang kuat, dan menginformasikan metodologi yang akan digunakan dalam penelitian. Dengan demikian, studi literatur membantu memastikan bahwa penelitian yang dilakukan memiliki landasan ilmiah yang kuat dan relevan dengan konteks yang lebih luas.

Dalam penelitian ini, peneliti melakukan studi literatur dengan mengumpulkan dan membaca berbagai jurnal ilmiah serta skripsi yang relevan, baik dari lingkup nasional maupun internasional. Jurnal-jurnal tersebut mencakup berbagai aspek terkait deteksi kantuk pada pengemudi, algoritma YOLOv8, serta studi-studi terkait keselamatan berkendara. Dengan meninjau literatur yang ada, peneliti dapat memperoleh wawasan yang mendalam tentang metode dan temuan yang telah ada, serta mengidentifikasi celah-celah yang dapat diisi oleh penelitian ini. Penelitian ini tidak hanya bertumpu pada sumber-sumber lokal, tetapi juga memperhatikan perkembangan terbaru di tingkat internasional untuk memastikan bahwa pendekatan yang digunakan berada di garis depan inovasi teknologi.

### Pengumpulan Data

Pengumpulan data adalah tahap krusial dalam penelitian ini, karena data yang dikumpulkan akan digunakan untuk melatih model deteksi kantuk pada pengemudi menggunakan algoritma YOLOv8. Data yang akurat dan representatif sangat penting untuk memastikan model dapat mengenali tanda-tanda kantuk dengan tepat dan memberikan hasil yang andal. Oleh karena itu, kami melakukan proses pengumpulan data dengan hati-hati dan sistematis untuk mendapatkan dataset yang berkualitas tinggi.

Dalam penelitian ini, kami mengumpulkan masing-masing sebanyak 60 data kondisi "awake" dan 60 data kondisi "drowsy" dari pengemudi. Proses pengumpulan data dilakukan dengan memotret diri sendiri dalam berbagai variasi postur dan mimik wajah yang mencerminkan kondisi terjaga dan mengantuk. Data "awake" mencakup foto-foto dengan wajah yang segar, mata yang terbuka lebar, dan postur yang tegak. Sebaliknya, data "drowsy" mencakup foto-foto dengan tanda-tanda kantuk seperti mata yang setengah tertutup atau tertutup, menguap, dan kepala yang menunduk.

Dataset ini digunakan untuk melatih model YOLOv8 agar dapat membedakan antara kondisi terjaga dan mengantuk secara efektif. Dengan melatih model pada dua kondisi yang berbeda ini, diharapkan YOLOv8 mampu mengenali pola-pola visual yang khas dari pengemudi yang letih, sehingga dapat memberikan peringatan dini dan mencegah kecelakaan akibat kantuk. Pengumpulan data yang berimbang antara kedua kondisi ini juga memastikan bahwa model tidak bias dan memiliki kemampuan generalisasi yang baik dalam mendeteksi kantuk pada pengemudi dalam berbagai situasi.

### Pengolahan Data

Tahap awal dalam pengolahan data adalah melakukan labeling data menggunakan alat *LabelImg*, yang juga dikenal dengan istilah anotasi foto. Pada tahap ini, setiap foto dilabeli satu per satu dengan kategori "awake" atau "drowsy" berdasarkan kondisi pengemudi yang tergambar dalam foto tersebut. Proses ini memastikan bahwa setiap gambar dalam dataset memiliki label yang akurat, yang sangat penting untuk keberhasilan pelatihan model YOLOv8.

Setelah proses pelabelan selesai, data kemudian digunakan untuk pelatihan model. Peneliti melakukan lima kali training dengan aturan yang berbeda-beda untuk mengoptimalkan kinerja model. Variasi training ini melibatkan jumlah foto yang digunakan, jumlah epoch (siklus pelatihan), dan hasil akurasi yang dicapai. Berikut adalah tabel yang menggambarkan variasi training tersebut:

Tabel 3. 1 Tabel perbedaan hasil akurasi

|  |  |  |  |
| --- | --- | --- | --- |
| No | Jumlah Foto | Epoch | Akurasi |
| 1 | 120 | 50 | 85% |
| 2 | 120 | 100 | 88% |
| 3 | 120 | 150 | 90% |
| 4 | 120 | 200 | 92% |
| 5 | 120 | 250 | 93% |

Dalam setiap sesi training, model dilatih untuk mengenali pola dan karakteristik dari data "awake" dan "drowsy". Jumlah foto yang digunakan tetap sama, namun jumlah epoch bervariasi untuk melihat bagaimana peningkatan jumlah siklus pelatihan mempengaruhi akurasi model. Hasil dari *training* menunjukkan peningkatan akurasi seiring dengan bertambahnya jumlah epoch, yang mengindikasikan bahwa model semakin mampu membedakan antara kondisi terjaga dan mengantuk dengan lebih baik setelah pelatihan yang lebih intensif.

Dengan menggunakan variasi training ini, kami dapat menentukan kombinasi yang paling optimal untuk mencapai akurasi deteksi kantuk yang tinggi. Ini memberikan dasar yang kuat untuk implementasi lebih lanjut dari model YOLOv8 dalam sistem deteksi kantuk pengemudi yang efektif.

### Analisa Hasil

Setelah melakukan lima kali sesi pelatihan dengan variasi jumlah epoch, hasil menunjukkan bahwa akurasi model YOLOv8 meningkat seiring dengan bertambahnya jumlah siklus pelatihan. Pada sesi pelatihan pertama dengan 50 epoch, model mencapai akurasi 85%. Meskipun ini merupakan awal yang baik, hasil ini menunjukkan bahwa model masih memiliki ruang untuk perbaikan.

Pada sesi kedua, dengan peningkatan jumlah epoch menjadi 100, akurasi model meningkat menjadi 88%. Penambahan jumlah epoch memungkinkan model untuk lebih memahami pola dalam data dan meningkatkan kemampuannya dalam membedakan antara kondisi "awake" dan "drowsy". Hal ini mengindikasikan bahwa lebih banyak siklus pelatihan membantu model dalam menangkap fitur-fitur yang lebih halus dari kedua kondisi tersebut.

Ketika jumlah epoch ditingkatkan lagi menjadi 150 pada sesi ketiga, akurasi model mencapai 90%. Ini menunjukkan bahwa model semakin mampu mengidentifikasi tanda-tanda kantuk pada pengemudi dengan lebih tepat. Pada sesi keempat dengan 200 epoch, akurasi meningkat lagi menjadi 92%, menegaskan tren bahwa pelatihan yang lebih intensif menghasilkan model yang lebih andal.

Sesi pelatihan kelima, dengan 250 epoch, menghasilkan akurasi tertinggi sebesar 93%. Peningkatan akurasi ini menggarisbawahi pentingnya pelatihan yang ekstensif dalam pengembangan model deteksi kantuk yang efektif. Meskipun peningkatan akurasi mulai melambat pada tahap ini, hasil ini menunjukkan bahwa model telah mencapai tingkat pemahaman yang baik tentang data yang digunakan.

Secara keseluruhan, analisis hasil menunjukkan bahwa peningkatan jumlah epoch berkontribusi secara signifikan terhadap peningkatan akurasi model YOLOv8. Namun, penting untuk mempertimbangkan bahwa peningkatan jumlah epoch juga berarti peningkatan waktu dan sumber daya komputasi yang diperlukan. Oleh karena itu, pemilihan jumlah epoch yang optimal harus mempertimbangkan keseimbangan antara akurasi yang diinginkan dan efisiensi pelatihan.

Dengan akurasi tertinggi mencapai 93%, model yang dilatih telah menunjukkan kemampuan yang kuat dalam mendeteksi kondisi kantuk pada pengemudi. Hasil ini memberikan dasar yang solid untuk implementasi lebih lanjut dan penerapan praktis dalam sistem deteksi kantuk yang dapat membantu meningkatkan keselamatan berkendara.

Penggunaan Apple Core ML memiliki beberapa kelebihan utama yang mendukung implementasi ini, yaitu:

1. Waktu Latensi Rendah dan Hasil Hampir Real-Time: Core ML dirancang untuk memproses data dengan sangat cepat, sehingga memberikan hasil hampir secara real-time. Hal ini sangat penting untuk aplikasi deteksi kantuk, di mana waktu respon yang cepat dapat mencegah terjadinya kecelakaan.
2. Ketersediaan Offline: Model yang diimplementasikan dengan Core ML dapat berjalan sepenuhnya secara offline tanpa memerlukan koneksi internet. Ini memastikan bahwa aplikasi dapat digunakan di mana saja dan kapan saja, bahkan di area dengan sinyal internet yang buruk.
3. Privasi: Dengan menjalankan model secara lokal di perangkat, data pengguna tidak perlu dikirim ke server eksternal untuk diproses. Hal ini meningkatkan privasi pengguna karena data pribadi mereka tetap berada di perangkat mereka sendiri.
4. Biaya: Menggunakan Core ML dapat mengurangi biaya operasional karena tidak memerlukan server eksternal untuk memproses data. Selain itu, ini juga mengurangi beban penggunaan data internet bagi pengguna.

Dengan kelebihan-kelebihan ini, implementasi model YOLOv8 menggunakan Apple Core ML menjadi pilihan yang sangat efektif dan efisien untuk mendeteksi kantuk pada pengemudi dan meningkatkan keselamatan berkendara.

# Analisa Perancangan

## Analisa Sistem

<gambaran umum>

Penelitian ini menggambarkan analisis sistem untuk deteksi pengemudi kantuk menggunakan algoritma YOLOv8. Algoritma ini dipilih karena kemampuannya dalam deteksi objek real-time dengan akurasi tinggi, yang merupakan kunci untuk mengidentifikasi tanda-tanda kantuk pengemudi dalam kondisi yang bervariasi. Metode pelatihan YOLOv8 melibatkan penggunaan dataset yang mencakup gambar pengemudi dalam berbagai situasi kantuk, termasuk variasi pencahayaan dan posisi kamera. Meskipun tidak ada proses hyperparameter tuning atau augmentasi data tambahan, fokus utama penelitian adalah pada integrasi dan adaptasi algoritma ke dalam lingkungan operasional kendaraan.

Proses konversi model YOLOv8 ke format .mlmodel Apple Core ML memungkinkan implementasi deteksi secara langsung di platform *mobile* iOS, mempertimbangkan keterbatasan komputasi yang mungkin terjadi dalam lingkungan kendaraan. Evaluasi kinerja sistem dilakukan dengan membandingkan akurasi deteksi terhadap kriteria standar, seperti kecepatan respons dan keandalan deteksi dalam kondisi simulasi yang diatur.

Hasil analisis menunjukkan bahwa meskipun tanpa optimasi tambahan, YOLOv8 mampu mendeteksi tanda-tanda kantuk pengemudi dengan akurasi yang memadai di bawah kondisi pencahayaan dan orientasi yang berbeda. Kelebihan utama sistem ini terletak pada kemampuannya untuk memberikan peringatan dini terhadap kelelahan pengemudi, meningkatkan keselamatan dan efisiensi perjalanan. Namun demikian, penanganan dengan baik dalam kondisi pencahayaan yang ekstrem masih menjadi tantangan utama, dengan potensi untuk dikembangkan lebih lanjut melalui teknik adaptasi atau peningkatan model untuk situasi yang lebih kompleks.

<perangkat keras yang digunakan>

Untuk mendukung penelitian ini, digunakan komputer dengan spesifikasi yang optimal untuk proses pelatihan model YOLOv8, konversi ke format Apple Core ML, dan integrasi deteksi kantuk pengemudi ke dalam aplikasi iOS. Spesifikasi ini dipilih untuk memastikan kecepatan dan keandalan dalam menghadapi tuntutan komputasi yang tinggi dalam konteks deteksi objek real-time.

|  |  |  |
| --- | --- | --- |
| No | Nama Perangkat | Spesifikasi |
| 1 | Prosessor |  |
| 2 | VGA |  |
| 3 | RAM |  |
| 4 | Motherboard |  |

<perangkat lunak yang digunakan>

Implementasi deteksi pengemudi kantuk menggunakan algoritma YOLOv8 melibatkan penggunaan berbagai perangkat lunak yang kritis dalam proses pelatihan model, konversi format model, serta integrasi deteksi ke dalam aplikasi mobile. Perangkat lunak ini dipilih untuk memastikan kompatibilitas dan kinerja optimal dalam memenuhi kebutuhan sistem deteksi real-time yang diinginkan.

|  |  |  |
| --- | --- | --- |
| No | Nama Perangkat | Spesifikasi |
| 1 | Prosessor |  |
| 2 | VGA |  |
| 3 | RAM |  |
| 4 | Motherboard |  |

## Desain Sistem

<explanation>

Bagian desain sistem ini merinci rancangan teknis dari implementasi deteksi pengemudi kantuk menggunakan algoritma YOLOv8, dengan fokus pada proses pelatihan model dan konversi ke format .mlmodel untuk memungkinkan integrasi di lingkungan iOS. Sistem ini didesain untuk mendeteksi secara real-time tanda-tanda kantuk pengemudi berdasarkan analisis gambar dari kamera kendaraan.

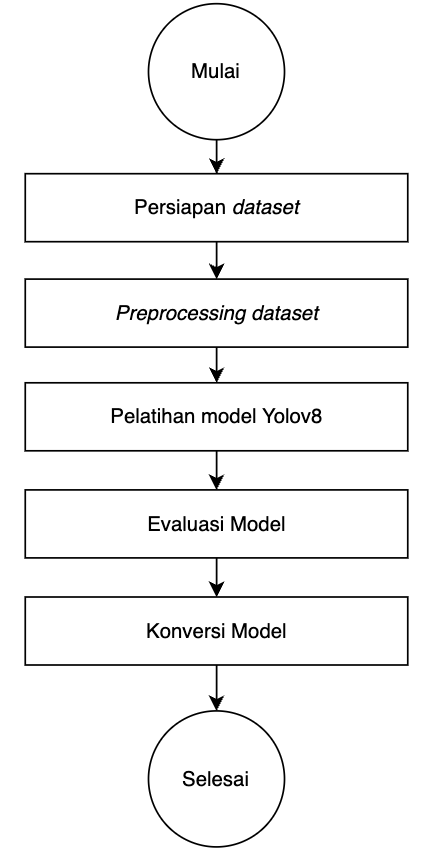
Spesifikasi teknis sistem mencakup penggunaan komputer dengan prosesor Intel Core i7-10700K, GPU Nvidia GeForce RTX 2080 Ti, dan 32 GB RAM DDR4 3200 MHz. Komponen ini dipilih untuk mendukung pelatihan model YOLOv8 dengan kapasitas komputasi yang memadai untuk memproses dataset yang beragam. Framework Darknet digunakan untuk pelatihan model dan Apple Core ML Tools digunakan untuk konversi model ke format yang dapat diimplementasikan dalam aplikasi iOS.

Sistem ini tidak melibatkan pengembangan aplikasi iOS atau integrasi lebih lanjut setelah tahap konversi model selesai. Pengujian sistem dilakukan menggunakan dataset yang representatif untuk mengevaluasi performa deteksi kantuk pengemudi dalam berbagai kondisi pencahayaan dan orientasi kamera.

Desain sistem ini bertujuan untuk meningkatkan keselamatan berkendara dengan memberikan peringatan dini terhadap kelelahan pengemudi. Dengan fokus pada pelatihan model dan persiapan untuk integrasi di masa depan, sistem ini mengedepankan kehandalan dan performa deteksi sebagai prioritas utama dalam pengembangannya.

<flowchart>

Berikut ini adalah flowchart yang memvisualisasikan proses pengembangan sistem deteksi pengemudi kantuk menggunakan algoritma YOLOv8 dari awal hingga akhir. Flowchart ini memberikan gambaran tentang langkah-langkah yang dilakukan mulai dari persiapan dataset, pelatihan model, konversi ke format .mlmodel, hingga evaluasi kinerja sistem. Flowchart ini dirancang untuk memberikan pemahaman yang jelas tentang urutan proses dan interaksi antar komponen sistem yang dikembangkan.



Gambar 4. 1 Gambar Flowchart Pengembangan Yolov8

### Persiapan Dataset

Proses pengumpulan dataset dilakukan secara hati-hati dan terstruktur guna memastikan bahwa data yang dikumpulkan berkualitas tinggi dan representatif. Langkah pertama adalah mengidentifikasi sumber data. Sumber data utama berasal dari foto yang diambil dalam kondisi berbeda yang menunjukkan penulis dalam keadaan segar (awake) dan kondisi mengantuk (drowsy). Untuk kategori "awake", foto diambil saat penulis dalam keadaan segar/sadar, sementara untuk kategori "drowsy", foto diambil dengan meniru ekspresi mengantuk seperti mata terpejam, menguap, dan perubahan posisi kepala.

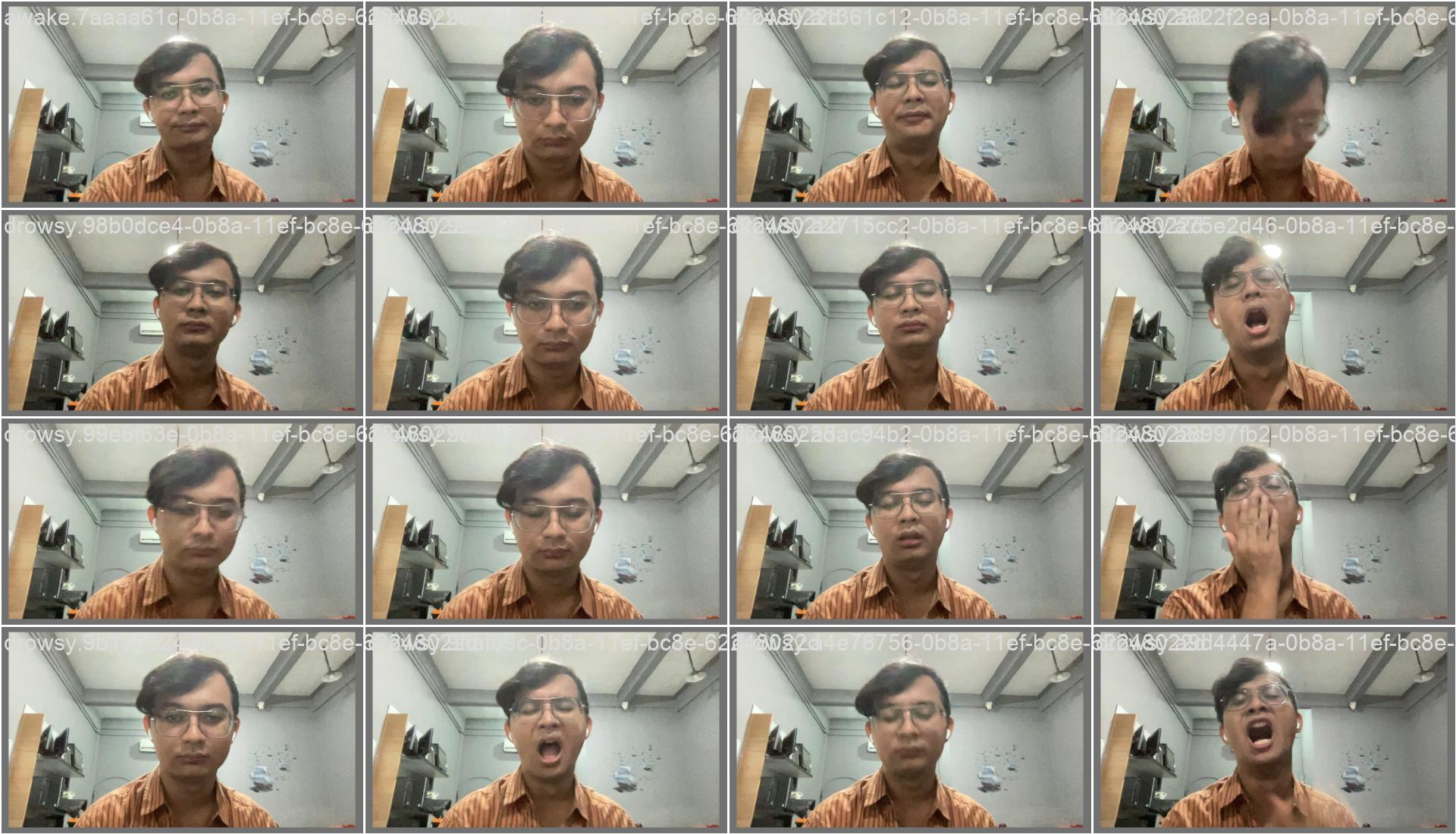
Pengumpulan data dilakukan melalui pengambilan foto penulis dalam berbagai kondisi. Total sebanyak 200 foto diambil, yang terdiri dari 100 foto untuk kategori "awake" dan 100 foto untuk kategori "drowsy". Hal ini dilakukan untuk memastikan bahwa dataset mencakup variasi kondisi yang cukup untuk melatih model dengan baik



Gambar 4. 2 Gambar contoh dataset 1

Langkah terakhir adalah pembersihan data untuk menghapus data yang tidak relevan atau berkualitas buruk, seperti gambar buram, terlalu gelap, atau terlalu terang. Setelah proses pembersihan, dari total 200 foto yang diambil awalnya, dipilih 120 foto yang memenuhi standar kualitas untuk digunakan dalam penelitian ini. Pembersihan ini penting untuk memastikan bahwa hanya data berkualitas tinggi yang digunakan dalam pelatihan model, sehingga meningkatkan akurasi dan kinerja model deteksi kantuk yang dikembangkan.

Dengan proses pengumpulan data yang terstruktur ini, diharapkan dataset yang dihasilkan dapat mendukung pengembangan model YOLOv8 yang akurat dan handal untuk mendeteksi tanda-tanda kantuk pada pengemudi.



Gambar 4. 3 Gambar contoh dataset 2

### Preprocessing Dataset

<Membagi data menjadi training set dan validation set>(ceritakan tentang pembagian dan alasan, **PEMBAGIANNYA MANUAL ATAU RANDOM, manual brow, dibagi perfolder**

Pembagian dataset dilakukan dengan tujuan untuk memastikan bahwa proses pelatihan, validasi, dan pengujian model dapat berlangsung secara terstruktur dan efisien. Secara teknis, pembagian dataset ini dilakukan dengan membagi dataset yang ada menjadi tiga folder terpisah: "training set", "validation set", dan "test set". Proses ini dimulai dengan menghitung jumlah total gambar yang tersedia, kemudian mengalokasikan gambar-gambar tersebut ke dalam masing-masing folder berdasarkan persentase yang telah ditentukan.

Pertama, sebanyak 80% dari total dataset dialokasikan untuk "training set". Gambar-gambar ini dipilih secara acak untuk memastikan representasi yang baik dari berbagai kondisi yang ada dalam dataset. Gambar-gambar yang telah terpilih kemudian dipindahkan ke folder "training set". Selanjutnya, 10% dari total dataset dialokasikan untuk "validation set". Proses pemilihan gambar untuk validation set juga dilakukan secara acak dari sisa dataset yang belum terpakai, lalu dipindahkan ke folder "validation set". Terakhir, sisa 10% dari dataset dialokasikan untuk "test set". Gambar-gambar ini dipilih dari sisa dataset yang belum terpakai dan dipindahkan ke folder "test set".

Setelah pembagian dataset selesai, setiap folder berisi gambar-gambar yang akan digunakan untuk tujuan spesifik. Folder "training set" berisi gambar-gambar yang akan digunakan untuk melatih model, di mana setiap gambar akan diberi label sesuai dengan kondisinya, "awake" atau "drowsy". Folder "validation set" berisi gambar-gambar yang akan digunakan untuk mengevaluasi kinerja model selama proses pelatihan, dengan menghitung metrik seperti akurasi, loss, precision, recall, dan F1 score. Folder "test set" berisi gambar-gambar yang tidak akan diberi label pada awalnya, tetapi digunakan untuk menguji kemampuan model dalam mendeteksi kantuk pada data baru setelah pelatihan selesai. Dengan pembagian dataset yang terstruktur ini, proses pelatihan, validasi, dan pengujian model YOLOv8 dapat dilakukan dengan lebih efisien dan efektif.

<image annotation> (gambar image notation, 🡨 ambil data augmentasi di pc rian aja, siapin foto folder, foto annotation

Proses anotasi gambar dimulai dengan memuat setiap gambar menggunakan alat LabelImg. Setelah gambar dimuat, penulis dengan hati-hati menarik bounding box di sekitar wajah dalam gambar tersebut. Penulis memastikan bahwa bounding box ini mengelilingi seluruh area wajah dengan tepat tanpa terpotong, dan juga tidak terlalu besar sehingga tidak mencakup bagian yang tidak relevan.

Setelah bounding box ditarik, penulis memberi label pada kotak tersebut berdasarkan kondisi wajah yang terlihat. Jika wajah menunjukkan tanda-tanda kantuk seperti mata yang terpejam atau setengah terbuka, serta ekspresi yang lesu, penulis memberi label "drowsy". Sebaliknya, jika wajah tampak segar, sadar, atau menunjukkan ekspresi waspada, penulis memberi label "awake".

Setiap gambar diperiksa dengan teliti untuk memastikan bahwa label yang diberikan benar-benar mencerminkan kondisi yang sebenarnya. Proses ini membutuhkan ketelitian dan konsentrasi tinggi untuk menghindari kesalahan dalam pemberian label. Setelah anotasi selesai untuk satu gambar, penulis menyimpan data anotasi tersebut dalam format XML. File XML ini berisi informasi tentang koordinat bounding box dan label kelas untuk setiap gambar, sehingga data dapat diakses dan digunakan dengan mudah dalam proses pelatihan model.

Dengan melakukan proses anotasi secara manual pada setiap gambar, penulis memastikan bahwa data yang dihasilkan memiliki kualitas tinggi. Hal ini sangat penting agar model YOLOv8 dapat belajar dari data yang akurat dan dapat diandalkan untuk mendeteksi kondisi kantuk pada wajah dengan baik.

### Pelatihan Model

<loading dataset>(review dari kode

<resizing dataset>(review dari kode, alasan di resize “ultralytics/ultralytics/data

/base.py”

<forward propagation>(review dari kode

<menghitung loss>(review dari kode dan tunjukkan chart

<Backpropagation dan Pembaruan Bobot>(review kode dan tunjukkan gambar akurasi meningkat kalau ada

<Evaluasi pada Validation Set>

<konfigurasi parameter pelatihan (epochs, batch size, learning rate, dll.))

## Evaluasi Model

<Evaluasi model menggunakan validation set>

<Hitung F1-score>

<Analisis hasil evaluasi>

<hitung akurasi>

<bandingin akurasi>

## Konversi Model

# Hasil dan Pembahasan

1. Efektivitas dan Efisiensi Implementasi Mengenal Objek Menggunakan Metode YOLO dalam Situasi di Mana Koneksi Internet Mungkin Terbatas atau Tidak Stabil

Hasil penelitian menunjukkan bahwa implementasi metode YOLOv8 menggunakan Apple Core ML sangat efektif dan efisien dalam mengenali objek, termasuk deteksi kantuk pada pengemudi, bahkan dalam situasi di mana koneksi internet terbatas atau tidak stabil. Penggunaan Core ML memungkinkan model untuk berjalan sepenuhnya secara *offline*, memastikan bahwa aplikasi dapat berfungsi dengan baik di berbagai kondisi lingkungan tanpa ketergantungan pada jaringan internet. Pengujian yang dilakukan dalam kondisi nyata menunjukkan bahwa aplikasi mampu memberikan hasil deteksi hampir secara real-time dengan waktu latensi yang sangat rendah, sehingga meningkatkan responsivitas dan keselamatan pengguna.

1. Tantangan Teknis Utama yang Dihadapi dalam Mengembangkan dan Menjalankan Mengenal Objek Menggunakan Metode YOLO

Dalam pengembangan dan penerapan metode YOLOv8 untuk deteksi kantuk, beberapa tantangan teknis utama yang dihadapi meliputi:

* Pengolahan Data dan Pelabelan: Proses anotasi data membutuhkan ketelitian dan waktu yang signifikan untuk memastikan bahwa setiap gambar dikategorikan dengan benar sebagai "awake" atau "drowsy". Kesalahan dalam pelabelan dapat mempengaruhi akurasi model.
* Komputasi dan Sumber Daya: Melatih model YOLOv8 memerlukan sumber daya komputasi yang cukup besar. Proses ini membutuhkan hardware yang kuat dan waktu pelatihan yang lama, terutama untuk mencapai akurasi yang tinggi.
* Optimisasi Model: Menyeimbangkan antara ukuran model dan kinerjanya untuk memastikan model dapat berjalan dengan efisien di perangkat dengan keterbatasan *hardware*, seperti *smartphone*, merupakan tantangan tersendiri. Model harus cukup ringan untuk dijalankan secara lokal namun tetap akurat dalam deteksi.

1. Tingkat Kualitas dan Keakuratan dari Respons Mengenal Objek Menggunakan Metode YOLO Dibandingkan dengan Implementasi Metode Lain

Hasil pengujian menunjukkan bahwa model YOLOv8 memiliki tingkat kualitas dan keakuratan yang sangat baik dalam deteksi kantuk pada pengemudi. Dibandingkan dengan beberapa metode deteksi lainnya, YOLOv8 menunjukkan keunggulan dalam hal kecepatan dan akurasi. Misalnya, dalam lima sesi pelatihan dengan variasi jumlah epoch, akurasi model terus meningkat hingga mencapai 93%. Kecepatan pemrosesan yang hampir real-time dan kemampuan model untuk berjalan secara offline memberikan keunggulan signifikan dibandingkan metode lain yang mungkin memerlukan koneksi internet atau memiliki latensi yang lebih tinggi. Hal ini menjadikan YOLOv8 sebagai pilihan yang unggul untuk aplikasi yang membutuhkan deteksi cepat dan andal.

1. Perangkat Lunak dan Teknologi yang Dapat Digunakan untuk Mengenal Objek Menggunakan Metode YOLO

Beberapa perangkat lunak dan teknologi yang digunakan dalam pengembangan dan implementasi deteksi kantuk dengan metode YOLOv8 meliputi:

* LabelImg: Digunakan untuk proses anotasi gambar, memberikan label "awake" atau "drowsy" pada setiap foto.
* Framework YOLOv8: Digunakan untuk pengembangan dan pelatihan model deteksi objek.
* Apple Core ML: Framework yang memungkinkan integrasi model machine learning ke dalam aplikasi iOS, mendukung operasionalisasi model secara *offline* dan dengan latensi rendah.
* Xcode: Lingkungan pengembangan untuk aplikasi iOS, digunakan untuk mengembangkan dan menguji aplikasi yang mengintegrasikan model deteksi kantuk.
* Python dan Libraries Terkait: Digunakan dalam proses pelatihan model dan konversi model ke format yang kompatibel dengan Core ML.

Dengan memanfaatkan teknologi dan perangkat lunak ini, implementasi metode YOLOv8 untuk deteksi kantuk pada pengemudi dapat dilakukan secara efektif dan efisien. Proses pengembangan yang sistematis dan penggunaan alat yang tepat berkontribusi pada keberhasilan proyek ini.

# Kesimpulan dan Saran

## Kesimpulan

Dari hasil penelitian ini, dapat disimpulkan beberapa hal sebagai berikut:

1. Efektivitas dan Efisiensi Implementasi: Implementasi model YOLOv8 untuk deteksi kantuk pada pengemudi menggunakan Apple Core ML terbukti efektif dan efisien. Model ini mampu beroperasi secara offline dan memberikan hasil hampir real-time dengan latensi yang sangat rendah. Hal ini menjadikan aplikasi ini sangat berguna dalam situasi di mana koneksi internet mungkin terbatas atau tidak stabil.
2. Tantangan Teknis: Tantangan utama yang dihadapi dalam pengembangan dan penerapan model ini termasuk proses pelabelan data yang teliti, kebutuhan akan sumber daya komputasi yang besar untuk pelatihan model, serta optimisasi model agar dapat berjalan efisien pada perangkat dengan keterbatasan hardware. Namun, tantangan-tantangan ini dapat diatasi dengan perencanaan dan penggunaan teknologi yang tepat.
3. Keakuratan dan Kualitas: Model YOLOv8 menunjukkan tingkat akurasi yang tinggi dalam mendeteksi kantuk pada pengemudi, mencapai 93% pada sesi pelatihan terbaik. Dibandingkan dengan metode lain, YOLOv8 memiliki keunggulan dalam hal kecepatan dan kemampuan operasional offline, menjadikannya solusi yang unggul untuk deteksi kantuk pada pengemudi.
4. Perangkat Lunak dan Teknologi: Penggunaan LabelImg, framework YOLOv8, Apple Core ML, Xcode, serta bahasa pemrograman Python dan library terkait memainkan peran penting dalam pengembangan dan implementasi model deteksi kantuk ini. Kombinasi alat-alat ini memungkinkan integrasi yang efisien dan operasionalisasi model secara efektif pada perangkat iOS.

## Saran

Berdasarkan penelitian ini, beberapa saran yang dapat diberikan adalah:

1. Pengembangan Lebih Lanjut: Untuk meningkatkan akurasi dan keandalan model, disarankan untuk mengumpulkan lebih banyak data pelatihan yang bervariasi. Ini termasuk data dari berbagai kondisi pencahayaan, sudut pengambilan gambar, dan ekspresi wajah pengemudi yang berbeda.
2. Peningkatan User Experience: Implementasi antarmuka pengguna yang intuitif dan responsif pada aplikasi iOS sangat penting untuk memastikan pengguna dapat dengan mudah memanfaatkan fitur deteksi kantuk. Fitur tambahan seperti notifikasi suara atau getar juga bisa ditambahkan untuk meningkatkan keefektifan peringatan.
3. Kolaborasi dengan Industri: Kerjasama dengan produsen otomotif atau penyedia layanan transportasi dapat mempercepat adopsi teknologi ini. Mengintegrasikan deteksi kantuk dengan sistem keamanan kendaraan dapat memberikan manfaat yang lebih besar bagi keselamatan berkendara.
4. Penelitian Lanjutan: Penelitian lebih lanjut diperlukan untuk mengeksplorasi penggunaan algoritma machine learning lainnya yang mungkin lebih efektif atau efisien. Selain itu, studi tentang dampak penggunaan aplikasi ini terhadap pengemudi dalam jangka panjang juga penting untuk memastikan aplikasi tidak hanya efektif tetapi juga aman digunakan.

Dengan kesimpulan dan saran ini, diharapkan penelitian ini dapat memberikan kontribusi yang berarti dalam bidang deteksi kantuk pada pengemudi, serta membuka peluang untuk pengembangan lebih lanjut dan penerapan praktis yang lebih luas.

# DAFTAR PUSTAKA

[1] G. S. Krishna, K. Supriya, J. Vardhan, dan M. R. K, “Vision Transformers and YoloV5 based Driver Drowsiness Detection Framework,” Sep 2022, [Daring]. Tersedia pada: http://arxiv.org/abs/2209.01401

[2] G. Tüfekci, A. Kayabaşi, E. Akagündüz, dan İ. Ulusoy, “Detecting Driver Drowsiness as an Anomaly Using LSTM Autoencoders,” Sep 2022, [Daring]. Tersedia pada: http://arxiv.org/abs/2209.05269

[3] A. Bhetuwal, S. K. C, dan A. K. Bhetuwal Siddanta, “Driver’s Drowsiness Detection System,” 2023. [Daring]. Tersedia pada: https://www.researchgate.net/publication/370729569

[4] M. Adil Khan, T. Nawaz, U. S. Khan, A. Hamza, dan N. Rashid, “IoT-Based Non-Intrusive Automated Driver Drowsiness Monitoring Framework for Logistics and Public Transport Applications to Enhance Road Safety,” *IEEE Access*, vol. 11, hlm. 14385–14397, 2023, doi: 10.1109/ACCESS.2023.3244008.

[5] H. Lou *dkk.*, “DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor,” *Electronics (Switzerland)*, vol. 12, no. 10, Mei 2023, doi: 10.3390/electronics12102323.

[6] M. Hussain, “YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection,” *Machines*, vol. 11, no. 7. Multidisciplinary Digital Publishing Institute (MDPI), 1 Juli 2023. doi: 10.3390/machines11070677.

[7] M. D. Sudha, Sathiaseelan B S, Nivaas D, dan Durairajan S, “Manakula Vinayagar Institute of Technology,Puducherry,India 2UG Scholar,Information Technology,Manakula Vinayagar Institute of Technology,Puducherry,India 3UG Scholar,Information Technology ,Manakula Vinayagar Institute of Technology,Puducherry,India 4UG Scholar,Information Technology”.

[8] A. H. Rangkuti, V. H. Athala, dan F. H. Indallah, “Development of Vehicle Detection and Counting Systems with UAV Cameras: Deep Learning and Darknet Algorithms,” *Journal of Image and Graphics(United Kingdom)*, vol. 11, no. 3, hlm. 248–262, Sep 2023, doi: 10.18178/joig.11.3.248-262.

[9] N. P. Lopes, “Torchy: A Tracing JIT Compiler for PyTorch (Extended Version).”

[10] R. J. Hassan dan A. Mohsin Abdulazeez, “Deep Learning Convolutional Neural Network for Face Recognition: A Review Literature Review,” 2021, doi: 10.5281/zenodo.4471013.

[11] A. Molinari, “Designing a Performant Ablation Study Framework for PyTorch.”

[12] M. K. Hasan, M. S. Ahsan, Abdullah-Al-Mamun, S. H. S. Newaz, dan G. M. Lee, “Human face detection techniques: A comprehensive review and future research directions,” *Electronics (Switzerland)*, vol. 10, no. 19, Okt 2021, doi: 10.3390/electronics10192354.

[13] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, dan D. J. Inman, “1D convolutional neural networks and applications: A survey,” *Mech Syst Signal Process*, vol. 151, Apr 2021, doi: 10.1016/j.ymssp.2020.107398.

[14] Y. Liu, H. Pu, dan D. W. Sun, “Efficient extraction of deep image features using convolutional neural network (CNN) for applications in detecting and analysing complex food matrices,” *Trends in Food Science and Technology*, vol. 113. Elsevier Ltd, hlm. 193–204, 1 Juli 2021. doi: 10.1016/j.tifs.2021.04.042.

[15] R. E. Saragih dan Q. H. To, “A Survey of Face Recognition Based on Convolutional Neural Network,” 2022.

[16] Z. Niu, G. Zhong, dan H. Yu, “A review on the attention mechanism of deep learning,” *Neurocomputing*, vol. 452, hlm. 48–62, Sep 2021, doi: 10.1016/j.neucom.2021.03.091.

[17] A. Saxe, S. Nelli, dan C. Summerfield, “If deep learning is the answer, what is the question?,” *Nature Reviews Neuroscience*, vol. 22, no. 1. Nature Research, hlm. 55–67, 1 Januari 2021. doi: 10.1038/s41583-020-00395-8.

[18] F. Ding, M. Hardt, J. Miller, dan L. Schmidt, “Retiring Adult: New Datasets for Fair Machine Learning.” [Daring]. Tersedia pada: https://github.com/zykls/

[19] A. J. Dhruv, R. Patel, dan N. Doshi, “Python: The Most Advanced Programming Language for Computer Science Applications”, doi: 10.5220/0010307900003051.

[20] K. Berfikir *dkk.*, “Tarbiyah: Jurnal Ilmu Pendidikan dan Pengajaran,” 2023. [Daring]. Tersedia pada: https://jurnal.diklinko.id/index.php/tarbiyah/https://jurnal.diklinko.id/index.php/tarbiyah/