MODEL ATENSI CITRA WAJAH AUDIENS BERBASIS *EDGE MACHINE LEARNING*

TESIS

Karya tulis sebagai salah satu syarat
untuk memperoleh gelar Magister dari
Institut Teknologi Bandung

Oleh
FITRI PUSPITASARI BUDIANA
NIM: 23220353
(Program Studi Magister Teknik Elektro)

INSTITUT TEKNOLOGI BANDUNG
Desember 2024

ABSTRAK

MODEL ATENSI CITRA WAJAH AUDIENS BERBASIS *EDGE MACHINE LEARNING*

Oleh
Fitri Puspitasari Budiana
NIM: 23220353
(Program Studi Magister Teknik Elektro)

Keterbatasan *edge device* akan kapasitas pemrosesan, penyimpanan dan memori, dapat membatasi kompleksitas komputasi pada komputasi tepi dalam mengekstraksi informasi langsung dari sumber data, yang akan diteruskan oleh *edge gateway* ke server atau *cloud* maupun *edge device* lainnya, melalui jaringan *Internet of Thing* (IoT) untuk pemrosesan data lebih lanjut. Sebuah model pada machine learning memiliki struktur matematis secara algoritmik, yang merupakan komputasi komplek dalam implementasi keceradan buatan untuk mengotomatisasi pemecahan masalah. Dalam penelitian ini, dengan metodologi *Design Research Methodology* (DRM) dan metode *Cross-Industry Standard Process for Data Mining* (CRISP-DM), diusulkan sebuah desain sistem komputasi *machine learning* di edge device, dengan menggunakan model *machine learning* berbasis *Convolution Neural Network* (CNN) yaitu arsitektur MobileNetV2. Proses *edge computing* dengan *machine learning* terjadi di ESP32-CAM yang terhubung dengan server melalui jaringan *Internet of Thing* (IoT). Komputasi machine learning pada ESP32-CAM memiliki keutamaan dalam proses mengklasifikasi image wajah audiens dalam momen Kegiatan Belajar Mengajar (KBM), yang tertangkap kamera ESP32-CAM untuk dikategorikan audiens tersebut dengan label fokus dan tidak-fokus. Data *image* beserta label tersebut, kemudian dikirim ke *Raspberry Pi.* Dataset yang digunakan dalam membangun model sebanyak 211 data training dan 63 data testing. Pengujian model dilakukan sebanyak 4 kali, dengan ukuran *batch* 64 dan jumlah *epoch* yang berbeda-beda. Pada *epoch* 55, Jumlah data uji sebanyak 63 berhasil mengklasifikan 45 dari 63 data uji citra dengan benar. Sehingga hasil tersebut didapatkan 71,43% untuk nilai akurasi, 69% untuk nilai AUC, 75% untuk nilai presisi, 76% untuk nilai recall, dan 76% untuk nilai F1 *score*.

Kata kunci: *Convolution Neural Network* (CNN), *Cross-Industry Standard Process for Data Mining (CRISP-DM), Design Research Methodology (DRM), Edge Computing, ESP32-CAM, Internet of Thing (IoT), Machine Learning, MobileNetV2, Raspberry Pi.*

ABSTRACT

AUDIENCE FACE IMAGE ATTENTION MODEL BASED ON EDGE MACHINE LEARNING

By
Fitri Puspitasari Budiana
NIM: 23220353
(Master's Program in Electrical Engineering)

The limitations of edge devices in processing capacity, storage, and memory can constrain the computational complexity of edge computing in extracting information directly from data sources. This information is then forwarded by the edge gateway to servers or the cloud, or to other edge devices, via the Internet of Things (IoT) network for further data processing. A model in machine learning has a mathematical structure algorithmically, which represents complex computations in the implementation of artificial intelligence to automate problem-solving. In this research, with the Design Research Methodology (DRM) Methodology and the Cross-Industry Standard Process for Data Mining (CRISP-DM) methods, The proposed design is a machine learning computing system on edge devices, using a Convolutional Neural Network (CNN) based model, specifically the MobileNetV2 architecture. The edge computing process with machine learning occurs on the ESP32-CAM, which is connected to a servers via the Internet of Things (IoT) network. Machine learning computation on the ESP32-CAM is particularly advantageous for the classification of audience facial images during teaching and learning activities, that Captured by the ESP32-CAM, the images are categorized with labels of focused and unfocused. The image data, including its labels, is then sent to Raspberry Pi. The dataset used in building the model consists of 211 training data and 63 testing data. Model testing was conducted 4 times, with a batch size of 64 and different number of epochs. The total number of test data is 63, with 43 out of the 63 test image data correctly classified. Thus, the results obtained are 71,43% for accuracy, 69% for AUC, 75% for precision, 76% for recall, and 76% for the F1 score.

Keywords: Convolution Neural Network (CNN), Cross-Industry Standard Process for Data Mining (CRISP-DM), Design Research Methodology (DRM), Edge Computing, ESP32-CAM, Internet of Thing (IoT), Machine Learning, MobileNetV2, Raspberry Pi.

MODEL ATENSI CITRA WAJAH AUDIENS BERBASIS *EDGE MACHINE LEARNING*

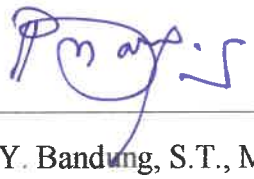
Oleh
Fitri Puspitasari Budiana
NIM: 23220353
(Program Studi Magister Teknik Elektro)

Institut Teknologi Bandung

Menyetujui
Tim Pembimbing

Tanggal 19 Desember 2024

Ketua



(Dr. Ir. Y. Bandung, S.T., M.T.)

PERNYATAAN ORISINALITAS

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tesis saya dengan judul Model Atensi Citra Wajah Audiens Berbasis *Edge Machine Learning* adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri. Semua referensi telah diacu dengan benar sesuai dengan kaidah penulisan ilmiah.

Bandung, 19 Desember 2024

Fitri Puspitasari Budiana
NIM. 23220353

PEDOMAN PENGGUNAAN TESIS

Tesis Magister yang tidak dipublikasikan terdaftar dan tersedia di Perpustakaan Institut Teknologi Bandung, dan terbuka untuk umum dengan ketentuan bahwa hak cipta ada pada penulis dengan mengikuti aturan HKI yang berlaku di Institut Teknologi Bandung. Referensi kepustakaan diperkenankan dicatat, tetapi pengutipan atau peringkasan hanya dapat dilakukan seizin penulis dan harus disertai dengan kaidah ilmiah untuk menyebutkan sumbernya.

Sitasi hasil penelitian Tesis ini dapat di tulis dalam bahasa Indonesia sebagai berikut:

Puspitasari Budiana, Fitri (2024): *Model Atensi Citra Wajah Audiens Berbasis Edge Machine Learning,* Tesis Program Magister, Institut Teknologi Bandung.

dan dalam bahasa Inggris sebagai berikut:

Puspitasari Budiana, Fitri (2024): *Audience Face Image Attention Model Based On Edge Machine Learning,* Master's Thesis, Institut Teknologi Bandung.

Memperbanyak atau menerbitkan sebagian atau seluruh tesis haruslah seizin Dekan Sekolah Pascasarjana, Institut Teknologi Bandung.

"...Sesungguhnya bersama kesulitan ada kemudahan, maka apabila engkau telah selesai maka (bekerjalah) hingga engkau letih dan hanya kepada Tuhanmu hendaknya engkau berharap". (Q.S Al Insyirah : 6-8)

Dipersembahkan kepada keluarga besarku tercinta yang senantiasa mendukung lahir dan batin.

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT yang telah memberikan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tesis yang berjudul "Model Atensi Citra Wajah Audiens Berbasis *Edge Machine Learning*".

Tesis ini merupakan salahsatu yang harus ditempuh untuk menyelesaikan Program Magister di Program Studi Magister Teknik Elektro, Sekolah Tinggi Elektro dan Informatika (STEI) Institut Teknologi Bandung. Proses penyusunan ini tidak terlepas dari bimbingan serta dukungan dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis mengucapkan banyak terima kasih kepada:

1. Bapak Dr. Ir. Y. Bandung, S.T., M.T selaku pembimbing dan dosen wali atas bimbingan dan arahan selama proses penelitian hingga terselesaikannya tesis ini.
2. Bapak Prof. Yusep Rosmansyah, ST., M.Sc, Ph.D, Bapak Dr.Ir. Albarda, MT dan Bapak Dr. Ir. Arry Akhmad Arman, MT selalu dosen penguji sidang tesis yang telah memberikan masukan dan saran untuk melengkapi kualitas tesis ini.
3. Orangtua dan Keluarga yang selalu mendukung dan berdoa.
4. Seluruh dosen STEI, staf dan karyawan STEI-ITB atas segala bantuan dan pembelajarannya selama ini.

Semoga tesis ini dapat memberikan manfaat bagi para pembaca.

Bandung, Desember 2024

Penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN DAN LAMBANG

SINGKATAN	NAMA	Pemakaian pertama kali pada halaman
AI	*Artificial Intelligence*	12
ANN	*Artificial Neural Network*	26
ARM	*Advanced RSIC Machine*	25
CNN	*Convolutional Neural Network*	2
CRISP-DM	*Cross-Industry Standard Process for Data Mining*	2
DRM	*Design Research Methodology*	2
DS-I	*Descriptive Study I*	35
DS-II	*Descriptive Study II*	35
IDE	*Integrated Development Environment*	24
IoT	*Internet of Things*	5
FNN	*Feedforward Neural Networks*	26
KBM	Kegiatan Belajar Mengajar	2
ML	*Machine Learning*	2
PCB	*Printed Circuit Board*	25
PIL	*Python Imaging Library*	53
PS	*Prescriptive Study*	35
RC	*Research Clarification*	35
RFID	*Radio-frequency identification*	17
RNN	*Recurrent Neural Networks*	26
SSID	*Service Set Identifier*	17
TIK	Teknologi Informasi dan Komunikasi	71

Bab I Pendahuluan

Pada bagian ini disampaikan tentang latar belakang penelitian, rumusan masalah, tujuan penelitian, batasan penelitian, kebaharuan dan kontribusi serta sistematika penulisan.

I.1 Latar Belakang

Melalui penggunaan arsitektur komputer canggih, *edge computing* menghindari masalah latensi dan *bandwidth* jaringan dengan memproses data secara efisien dan cepat di dekat sumbernya. *Edge computing* memperpendek waktu yang dibutuhkan pengguna untuk mengirimkan data sekaligus meringankan pusat komputasi awan dari sebagian beban pemrosesan dan transmisi dengan mengalihkan kemampuan pemrosesan ke tepi jaringan. Penundaan akses dapat menjadi hambatan, mengaburkan keuntungan dari edge computing, terutama bagi bisnis yang memerlukan sejumlah besar data. *Edge computing* menghadirkan banyak tantangan, termasuk keamanan, kehilangan data, serta biaya investasi dan pemeliharaan (Jebamani, 2021).

Internet of Things (IoT) telah mengalami perkembangan yang sangat pesat dalam beberapa tahun terakhir, menghubungkan miliaran perangkat dan menghasilkan volume data yang sangat besar dengan kecepatan tinggi. Meskipun IoT menawarkan berbagai keuntungan, tantangan utama dalam hal keamanan, privasi, dan pemrosesan data masih terus menjadi perhatian utama. Pada awalnya, masalah keamanan IoT terutama berfokus pada otentikasi dan otorisasi perangkat *edge* yang memiliki sumber daya terbatas. Banyak solusi yang diusulkan di masa lalu mengandalkan peningkatan kapasitas komputasi, penyimpanan, dan daya pada perangkat *edge*, namun solusi ini sering kali tidak praktis karena keterbatasan ukuran perangkat atau biaya yang tinggi (Zareen, 2019).

Dengan meningkatnya volume data yang dihasilkan oleh sensor dan aktuator dalam IoT, pemrosesan data di *cloud* tradisional menghadapi kendala serius terkait bandwidth jaringan dan latensi komunikasi. *Edge computing* telah muncul sebagai

solusi yang menjanjikan untuk mengatasi tantangan ini dengan membawa kapasitas komputasi lebih dekat ke sumber data (Premsankar et al., 2018). Penelitian terbaru menggaris bawahi bahwa *edge computing* dapat mengurangi latensi dan meningkatkan kinerja aplikasi yang memerlukan respons cepat, seperti gaming mobile dan aplikasi *virtual reality* (Pan & McElhannon, 2018).

Arsitektur *edge computing* yang mendekatkan sumber daya komputasi ke perangkat IoT tidak hanya membantu mengatasi masalah latensi tetapi juga mengurangi beban pada pusat data *cloud* (F. Liu et al., 2019a).

Dalam konteks IoT, *machine learning* (ML) dan AI menjadi kunci untuk meningkatkan efisiensi dan efektivitas sistem edge (Kong & Li, 2017). *Machine learning* dapat digunakan untuk berbagai aplikasi, termasuk deteksi masker wajah pada perangkat *edge* selama pandemi Covid-19 dan untuk mendeteksi status pembelajaran siswa dalam konteks pendidikan online (Jovovic et al., 2022).

Penggunaan *machine learning* di *edge computing* memungkinkan pemrosesan data secara lokal pada perangkat dengan keterbatasan sumber daya (Manjunath et al., 2021), mengurangi kebutuhan untuk mengirimkan data ke *cloud* dan mengatasi masalah latensi serta privasi (Samie et al., 2019). *Edge computing* dan *machine learning* menawarkan potensi besar untuk mengatasi tantangan dalam ekosistem IoT yang berkembang. Dengan arsitektur yang efisien dan teknologi baru, seperti *network function virtualization* dan *software defined networking*, *edge computing* dapat mendukung aplikasi IoT yang memerlukan latensi rendah dan pemrosesan data yang cepat (Li et al., 2022) (Cao et al., 2020).

Vaishnavi C dan Suja Palaniswamy (2022) berfokus pada emosi dari video kelas online menggunakan teknik *meta-learning*, yang meningkatkan kemampuan model untuk belajar dari data baru dengan lebih cepat dan efesien. Meta-learning memungkinkan model untuk beradaptasi dengan variasi emosi dan ekspresi yyang berbeda dengan lebih baik. Penelitian ini menggunakan kombinasi jaringan saraf konvolusional berbasis lapisan (*layer-based Convolutional Neural Networks*, CNN) diusulkan, yang disebut SNSER (*Siamese Network for Student Emotion Recognition Model*), dilatih menggunakan dataset CAFE (*Children Affective Facial*

Expression), yang mencakup tujuh emosi dasar yaitu Neutral, Angry, Surprise, Sad, Fear, dan Disgust. Model ini mencapai akurasi 80% dalam mengenali emosi (Vaishnavi & Palaniswamy, 2022).

Penelitian lain yang dilakukan oleh Haipeng Zeng, et.al (2021) menganalisis emosi siswa dari video kelas yang dapat membantu guru dan orang tua memahami keterlibatan siswa di kelas. EmotionCues, sebuah sistem analitik visual untuk menganalisis video kelas dari sudut pandang ringkasan dan analisis mendetail emosi, yang menggabungkan algoritma pengenalan emosi dengan visualisasi. Data yang digunakan berupa data video yang dikumpulkan dari berbagai taman kanak-kanak. Setiap video berdurasi 10 menit dengan resolusi 1920 x 1080 dan 30 fps. Sehingga total frame 18.000. Ada beberapa algoritma yang digunakan, pertama adalah algoritma MTCNN untuk face detection yang melibatkan library openCV dan dlib. serta Facenet untuk pengenalan wajah melalui vektorisasi gambar. Model CNN (ResNet-50) dilatih dengan dataset FER 2013 untuk pengenalan ekspresi wajah dengan akurasi sekitar 71,2%. Hasil evaluasi menunjukkan akurasi deteksi wajah sebesar 95,9% pada video taman kanak-kanak dan 96,3% pada video seminar. Akurasi pengenalan emosi mencapai 64,8% pada video taman kanak-kanak dan 68,5% pada video seminar. Meskipun terdapat tantangan dalam menangkap emosi siswa dalam situasi tertentu, sistem ini dilengkapi dengan fitur untuk menangani emosi yang tidak terdeteksi dan memberikan petunjuk lebih lanjut tentang kondisi siswa (Zeng et al., 2021).

Bo Jiang et.al (2019) melakuakan penelitian analisis otomatis perilaku di dalam kelas dengan mengusulkan model konsentrasi kelas berdasarkan informasi pose dan posisi kepala yang dikumpulkan oleh jaringan saraf konvolusional. Dengan dataset yang dikumpulkan sendiri, penelitian ini memiliki akurasi deteksi kepala sebesar 97,4% dan akurasi posisi kepala sebesar 94,6% (Jiang et al., 2019).

Sadia Afroze et.al (2022) melakukan penelitian berupa kerangka kerja berbasis visi untuk mendeteksi *focus of attention* (FOA) dari seseorang dengan menggunakan sembila pose kepala yang terdiri dari empat modul utama: *face detection and facial key-point selection* (FDKPSM), *head pose classification* (HPCM), *object*

localization and classification (OLCM), dan *focus of attention estimation* (FoAEM). FDKPSM menggunakan kerangka kerja *Multi-task Cascaded Neural Network* (MTCNN) untuk mendeteksi pose kepala, dan HPCM mengklasifikasikannya menjadi sembilan kelas menggunakan ResNet18. Untuk memperkirakan FOA, FoAEM menggunakan Algoritma pemetaan (EFoA) yang mengintegrasikan pose kepala pada objek yang difokuskan. Hasil eksperimen menunjukkan bahwa model yang diusulkan mengungguli model *deep learning* lainnya dengan mencapai akurasi tertinggi pada tiga dataset: BIWI-M (96,97%), Pointing'04-M (96,04%), dan HPoD 9 (98,99%). Model visual focus of attention mendapatkan akurasi 94,12% pada skenario multi-objek (Afroze et al., 2022).

Penelitian lain juga dilakukan oleh Partha Chakraborty et.al (2021) melakukan penelitian berbagai pendekatan *machine learning* untuk memprediksi tingkat visual focus of attention manusia. Dengan menggunakan data yang dikumpulkan dari laporan survei (data lingkungan) dan data eyeball (pergerakan mata, waktu membaca, dan gerakan kepala) saat membaca artikel, sebuah dataset dengan tingkat perhatian setiap peserta dibentuk. Delapan classifier yang berbeda dilatih, yaitu: *Logistic Regression, Support Vector Machine* (SVM), *Decision Tree, K-Nearest Neighbor* (KNN), *AdaBoost, Multilayer Perceptron* (MLP), *Extra Tree Classifier*, dan *Voting Classifier* untuk mengklasifikasikan tingkat perhatian peserta ke dalam tiga kelas: *High*, *Average*, dan *Low*. *Logistic Regression* mencapai akurasi tertinggi sebesar 96%, mengungguli yang lain dalam memprediksi kelas-kelas ini. Voting Classifier berbobot gabungan mencapai akurasi 95% (Chakraborty et al., 2021).

Liu, T., et al. (2021) dalam penelitiannya menggunakan gambar pose kepala inframerah yang serupa dengan dataset IRHPE sebagai dataset primer, serta CAS-PEAL-R1 dan Pointing'04 sebagai dataset sekunder. Penentuan label gambar pose menggunakan metode Gaussian nonuniform dan neural network. Hasil penelitian menunjukkan bahwa NGDNet secara signifikan mengungguli algoritma konvensional dalam hal estimasi pose kepala inframerah. Model NGDNet mencapai akurasi 77,39% pada dataset IRHPE, 99,08% pada dataset CAS-PEAL-R1, dan 87,41% pada dataset Pointing'04 (T. Liu et al., 2021).

Penelitian lain yang dilakukan oleh Tripti Singh et al. (2021) mengeksplorasi penggunaan berbagai teknik seperti regresi elastic net, CPAM, dan DNN untuk menangani dataset besar, pemrosesan data waktu nyata, variasi pose, oklusi parsial, dan ekspresi wajah yang beragam. Model yang diusulkan dapat mencapai kesalahan absolut rata-rata (MAE) sebesar 3° atau kurang. Evaluasi dilakukan pada tiga dataset standar yaitu 300W-LP (300W across large poses), AFLW2000 (Annotated Facial Landmarks in the Wild), dan NIMH-ChEFS. Hasil evaluasi menunjukkan bahwa kinerja model ini sebanding dengan metodologi terbaru seperti AADL, JFA, dan RAFA-Net, dengan MAE mencapai 6° (Singh et al., 2021).

Sadia Afroze, et al. (2020) dalam penelitiannya menggunakan data latih yang terdiri dari 7 jam video dengan 435.000 frame dan 9 pose kepala, serta data uji yang terdiri dari 81.000 frame. Metode yang digunakan meliputi Multi-task *Cascaded Convolutional Networks* (MTCNN), *Intersection over Union* (IoU), *Proposal Network* (P-Net), dan *You Only Look Once* (YOLO). Model klasifikasi yang dikembangkan mencapai akurasi 97,00% pada set uji. Selain itu, dalam skenario multi-objek, akurasi model mendekati 95,00% (Afroze & Hoque, 2020).

Penelitian lain yang dilakukan oleh Chongyang Bai et al. (2019) mengeksplorasi model ICAF (*Iterative Collective Attention Focus*) untuk klasifikasi kolektif menggunakan dataset baru yang terdiri dari 5 video (35 orang, 109 menit, 7604 label) dari permainan Resistance. Dalam model ini, setiap orang dimodelkan menggunakan pengklasifikasi terpisah, di mana prediksi dari pengklasifikasi orang lain digunakan sebagai input untuk setiap pengklasifikasi individu, menggabungkan ketergantungan perilaku. ICAF menunjukkan peningkatan akurasi sebesar 1%-5% dibandingkan baseline terkuat dalam memprediksi fokus perhatian visual (Bai et al., 2019).

Patricia Goldberg, et al. (2019) melakukan penelitian menggunakan data video rekaman kelas yang dianotasi secara manual dengan mengkode perilaku siswa setiap detik menggunakan software CARMA dengan skala -2 hingga +2. Metode

otomatis yang digunakan dalam penelitian ini adalah pendekatan vision-based machine learning dengan fitur seperti pose kepala, arah pandangan, dan ekspresi wajah. Algoritma yang digunakan adalah *Support Vector Regression* (SVR) untuk mengestimasi intensitas keterlibatan siswa. Hasil penelitian menunjukkan bahwa kombinasi pose kepala, arah pandangan, dan ekspresi wajah memiliki korelasi yang tinggi dengan anotasi manual (r = 0,61). Akurasi prediksi keterlibatan yang dihitung dengan mean squared error (MSE) dan koefisien korelasi Pearson bervariasi, dengan model terbaik yang menggabungkan sinkroni antar siswa menunjukkan peningkatan 9% dalam korelasi dengan anotasi manual (Goldberg et al., 2021).

Benoit Massé, Silèye Ba, dan Radu Horaud (2017) menggunakan dua dataset yang tersedia secara publik yang mengandung interaksi manusia-robot dan manusia-manusia multi-partai. Mereka menerapkan model ruang keadaan switching Bayesian dengan variabel laten untuk *Visual Focus of Attention* (VFOA) dan tatapan mata. Penelitian ini memperkenalkan prosedur pembelajaran yang dapat ditangani serta algoritma efisien untuk melacak tatapan dan fokus visual (Massé et al., 2017).

Dalam sebuah penelitian, dirancang sebuah perangkat yang mengubah pintu biasa menjadi pintu pintar yang hanya akan memberikan akses kepada orang yang mengenakan masker. TensorFlow dan OpenCV pada Raspberry Pi digunakan untuk mendeteksi apakah seseorang memakai masker atau tidak. Jarak efektif perangkat ini adalah 2,7 m dengan akurasi mencapai 98,11%. Penelitian lain juga dilakukan oleh Mohd Nahdir, et.al (2021) dengan kemajuan algoritma deep learning seperti Convolutional Neural Network (CNN), akurasi sistem semakin meningkat. Model hybrid CNN - KNN dapat meningkatkan akurasi FER, di mana CNN digunakan untuk ekstraksi fitur, dan KNN untuk pengenalan ekspresi. Dengan memanfaatkan transfer learning dan meggunakan EfficienNet-Lite, model ini mencapai akurasi sebesar 75,26% pada perangkat Raspberry Pi dengan keterbatasan sumber daya komputasi dan memori (Ab Wahab et al., 2021).

Mokhamad Arfan dan Yoanes (2022) merancang Protokol Constrained Application Procotocol (CoAP) untuk komunikasi pada perangkat embedded dengan sumber daya terbatas, bekerja di atas User Datagram Protocol (UDP) dengan overhead lebih rendah dibandingkan Transmission Control Protocol (TCP). Dalam penelitian ini, CoAP dievaluasi untuk komunikasi data multimedia pada jaringan sensor multimedia nirkabel (WMSN). Pengukurang throughput dan jumlah kejadian retransmisi menunjukkan bahwa penambahan perangkat sensor menyebabkan penurunan throughput rata - rata secara linear (Wicaksono & Bandung, 2022).

Sebuah arsitektur IoT 3-lapis dikembangkan untuk mendeteksi dan mengidentifikasi individu dengan penyakit infeksius demam di area publik dalam ruangan. Sistem ini terdiri dari lapisan sensing, edge computing, dan clous computing. Modul sensor menangkap gambar wajah dan suhu tubuh, yang ditransmisikan ke perangkat edge melalui MQTT untuk identifikasi menggunakan teknik machine learning. Data selanjutnya diteruskan ke server cloud untuk penyimpanan dan analisis lebih lanjut (Gianto et al., 2023).

Vidia Kamath, et al. (2024) membandingkan berbagai arsitektur CNN, termasuk VGG16-SSD yang dilatih menggunakan dataset MOD-2022 untuk deteksi dan pelacakan objek. Model VGG16-SSD, yang dilatih menggunakan dataset tersebut, menunjukkan kinerja yang menurun pada gambar berkualitas rendah, namun tetap mampu mendeteksi dan melacak objek dengan akurasi yang dapat diterima, bahkan pada perangkat terbatas seperti Raspberry Pi. Penelitian ini juga menunjukkan bahwa dengan membuka lapisan terakhir hingga 20 lapisan dari arsitektur model, model dapat disesuaikan dengan set gambar IoT baru, tergantung pada arsitektur CNN yang digunakan. Selain itu, teknik optimisasi seperti kuantisasi terbukti memperkecil ukuran model hingga 2× atau 3×, yang menghasilkan jejak memori lebih ringan, waktu eksekusi lebih cepat, dan konsumsi daya lebih rendah. Bahkan, perangkat seperti Coral Dev Board mampu meningkatkan proses inferensi hingga 100×, sementara arsitektur model EfficientNet dapat mempertahankan akurasi klasifikasi secara efisien (Kamath et al., 2024).

Paul D. Rosero-Montalvo, et al. (2024) melakukan penelitian menggunakan lima dataset berbeda untuk tugas deteksi dan klasifikasi, dengan total 122.124 gambar. Penelitian ini menggunakan metode optimisasi arsitektur CNN dengan TensorFlow Lite (Tf-lite) dan kuantisasi (IQ) untuk perangkat edge. Model Tf-lite memanfaatkan semua inti CPU di bawah 80%, dengan VGG-16 mencapai penggunaan 100% pada semua inti. Sebaliknya, model kuantisasi hanya menggunakan satu inti CPU lebih dari 90% dan dua inti lainnya di bawah 20%. Perbedaan ini menunjukkan variasi dalam penggunaan inti CPU dan konsumsi daya antara kedua metode tersebut (Rosero-Montalvo et al., 2024).

Neha Patil, et al. (2017) melakukan penelitian dengan menggunakan gambar gerakan atau gestur yang diambil menggunakan webcam standar. Metode yang diterapkan melibatkan penggunaan Raspberry Pi yang dilengkapi dengan OpenCV untuk pemrosesan gambar dan deteksi gerakan atau gestur. Hasil penelitian menunjukkan bahwa sistem ini mampu memantau dan memberikan peringatan secara efektif saat gerakan atau gestur terdeteksi. Selain itu, sistem dapat mengirimkan gambar yang terdeteksi ke server cloud atau menyimpannya secara lokal jika koneksi ke server cloud tidak tersedia, sambil mempertimbangkan efisiensi konsumsi daya. Hal ini menunjukkan bahwa solusi yang diusulkan tidak hanya efektif dalam deteksi dan pemberian peringatan, tetapi juga efisien dalam penggunaan sumber daya, menjadikannya cocok untuk aplikasi berbasis IoT (Patil et al., 2017).

Zehra Ozkan, et al. (2021) melakukan penelitian dengan menggunakan 3948 gambar yang diambil dari berbagai kondisi cuaca, lingkungan, dan waktu untuk melatih model deteksi dan pengenalan ancaman. Metode yang digunakan adalah pembelajaran mesin dengan jaringan saraf konvolusional (CNN), di mana dua arsitektur, Faster-RCNN dan SSD MobileNet V2, dipilih untuk membandingkan akurasi deteksi. Hasil penelitian menunjukkan bahwa deteksi dan pengenalan objek dengan Faster-RCNN mencapai akurasi 91%, sedangkan SSD MobileNet V2 mencapai akurasi 88% (Ozkan et al., 2021).

Penelitian ini bertujuan untuk menerapkan desain model yang mampu mengklasifikasi image dalam kategori image wajah yang menandakan fokus dan tidak fokus dengan memanfaatkan teknologi edge computing dan machine learning. Model ini dirancang untuk diintegrasikan dengan perangkat IoT seperti ESP32-CAM dan Raspberry Pi, dengan tujuan utama untuk meningkatkan pengalaman pengguna dalam aplikasi konferensi video dengan memastikan bahwa gambar yang dihasilkan selalu dalam kondisi fokus optimal.

Model *machine learning* dengan arsitektur MobileNetV2 dirancang untuk efisiensi komputasi yang tinggi, yang berarti model ini dapat dijalankan dengan cepat bahkan pada perangkat dengan daya komputasi rendah seperti smartphone dan perangkat IoT. Karena efisiensi komputasi dan ukuran model yang kecil, MobileNetV2 mampu melakukan *inference* (proses prediksi) dengan cepat. Ini sangat penting untuk aplikasi *real-time* yang membutuhkan respon cepat, deteksi objek langsung dari kamera sebagai *edge device* untuk melakukan proses komputasi *edge computing*.

Kebaharuan dan Kontirbusi dari penelitian ini terletak pada pengembangan metode inovatif untuk integrasi *edge computing* dengan perangkat IoT, penerapan machine learning untuk deteksi fokus gambar, serta peningkatan efisiensi aplikasi IoT yang memerlukan pemrosesan data real-time. Penelitian ini juga memberikan gambaran menyeluruh mengenai perkembangan terkini dalam *edge computing* dan *machine learning*, serta bagaimana integrasi teknologi ini dapat meningkatkan efisiensi dan efektivitas sistem IoT secara keseluruhan. Dengan demikian Penelitian ini, mengusulkan Model Atensi Citra Wajah Audiens Berbasis *Edge Machine Learning*.

I.2 Rumusan Masalah

Berdasarkan latar belakang yang sudah diuraikan sebelumnya, didapatkan rumusan masalah yang akan diulas adalah Bagaimana mengimplementasikan *machine learning* untuk deteksi kefokusan audiens berdasarkan wajah pada peralatan *edge computing* berbasis *Internet of Thing* dalam Kegiatan Belajar Mengajar (KBM).

I.3 Tujuan Penelitian

Penelitian ini bertujuan untuk membangun sistem deteksi fokus dan tidak fokus pada image wajah berbasis *edge computing* dan *machine learning* dengan menggunakan *Convolutional Neural Networks* (CNN) dengan Model Arsitektur MobileNetV2 yang diterapkan menggunakan ESP32-CAM dan Raspberry Pi, guna mendukung peningkatan kualitas Kegiatan Belajar Mengajar

I.4 Batasan Penelitian

Penelitian ini dibatasi pada peralatan hardware maupun *software* sebagai berikut

a. *Edge Impulse* yang berfungsi untuk mentraining model
b. Proses komputasi *machine learning* dengan arsitektur MobileNetV2 di ESP32-CAM
c. Multimedia pada penelitian adalah dalam bentuk gambar dan bertipe *.JPG
d. Perangkat Multimedia IoT yang berfungsi sebagai node sensor adalah ESP32-CAM dan Raspberry Pi.
e. Batasan dalam deteksi kondisi fokus dan tidak fokus adalah bahwa model akan menganggap seseorang tidak fokus ketika kepala mereka menghadap ke atas atau ke bawah (sudut vertikal), atau menengok ke kanan atau kiri (sudut horizontal).

I.5 Hipotesis

Berdasarkan literatur yang ada, beberapa penelitian telah berhasil dalam hal-hal di bawah ini:

a) Mendesain model *machine learning*
b) Menentukan dataset training
c) Intruksi-intruksi model *machine learning* dapat disimpan di *Edge Device*
d) ESP32-CAM dapat mengcapture *image*
e) Mengklasifikasi data uji dengan *Convolutional Neural Networks* (CNN) di Edge Device ESP32-CAM
f) *Edge Device* ESP32-CAM dapat mengirim data ke server Raspberry Pi
g) Menyimpan hasil klasifikasi di server melalui jaringan IoT

Dengan demikian, penelitian ini berhipotesis yaitu: membangun sistem deteksi fokus dan tidak fokus pada image wajah berbasis machine learning dengan menggunakan *Convolutional Neural Networks* (CNN) dengan Model Arsitektur MobileNetV2 yang diterapkan menggunakan ESP32-CAM, dalam lingkungan edge computing dan server, guna mendukung peningkatan kualitas Kegiatan Belajar Mengajar (KBM).

I.6 Kebaharuan dan Kontribusi

Kebaharuan pada penelitian ini berupa Penerapan MobileNetV2 pada ESP32-CAM. MobileNetV2 pada umumnya sering digunakan pada perangkat ponsel karena model ini dirancang untuk pengolahan gambar yang efisien, namun ponsel memiliki kapasitas komputasi yang jauh lebih besar dibandingkan dengan ESP32-CAM. Penggunaan MobileNetV2 pada perangkat dengan keterbatasan seperti ini, yang biasanya tidak cukup kuat untuk menjalankan model deep learning besar, merupakan hal yang baru dan menarik. Dengan optimasi model yang tepat (misalnya *quantization*, *pruning*, atau *transfer learning*), MobileNetV2 bisa dioptimalkan agar dapat berjalan di ESP32-CAM, memberikan kebaruan dalam aplikasi edge machine learning yang hemat sumber daya. Kontribusi pada penelitian ini yaitu:

a. Peningkatan Akurasi dan Efisiensi pada *Edge Device*

   Model ini memungkinkan pengenalan wajah yang lebih akurat dan efisien, meskipun dijalankan pada perangkat dengan keterbatasan memori dan daya. Hal ini membuka peluang untuk aplikasi pengenalan wajah di berbagai perangkat IoT dan sistem pengawasan berbasis edge.

b. Penerapan Klasifikasi Citra pada Perangkat *Edge* Terbatas

   Penelitian ini memberikan kontribusi besar dengan mengimplementasikan klasifikasi citra wajah menggunakan MobileNetV2 pada ESP32-CAM, memperkenalkan pendekatan baru untuk pengenalan wajah pada perangkat *edge* dengan sumber daya terbatas, yang dapat diterapkan pada berbagai aplikasi IoT dan pengawasan berbasis pengenalan wajah.

I.7 Sistematika Penulisan

Bab I Pendahuluan

Bab ini berisi penjelasan tentang latar belakang, rumusan masalah, tujuan penelitain, batasan penelitian, keluaran penelitian, metodologi penelitian dan sistematika penulisan laporan.

Bab II Tinjuan pusataka

Bab ini penjelasan tentang teori-teori yang berkaitan dengan topik/masalah penelitian diantaranya meliputi tinjauan *Internet of Things* (IoT), *Machine Learning*, dan *State of the Art* dari penelitian-penelitian terdahulu dan relevan.

Bab III Metodologi Penelitian

Bab ini berisi penjelasan tentang metodologi desain penelitian, yaitu menggunakan metoda *Design Research Methodology* (DRM) yang sangat membantu merencanakan dan mengimplementasikan penelitian desain.

Bab IV Analisis Model

Bab ini berisi penjelasan tentang detail tahapan pada model proses data, yang meliputi; *Business understanding, Data understanding, Data preparation,* dan *Modeling.*

Bab V Perancangan dan Implementasi

Bab ini berisi penjelasan tentang detail tahapan pada model proses data, yang meliputi Skenario Pengujian dan *Deployment*.

Bab VI Penutup

Bab ini berisi kesimpulan yang dapat diberikan berdasarkan hasil penelitian ini. Sedangkan saran untuk menyampaikan bahasan yang belum tuntas dan bisa dijadikan untuk penelitian berikutnya.

Bab II Tinjauan Pustaka

Pada Bab ini, akan dipaparkan berbagai teori sebagai dasar gagasan dari berbagai penelitian terdahulu yang relevan terhadap topik penelitian sebagai landasam berpikir untuk melaksanakan penelitian sehingga diperoleh pembuktian yang valid atas hipotesa awal penelitian.

II.1 Kerangka Konseptual

Pada bagian ini disampaikan referensi yang digunakan dalam penelitian melalui referensi paper, *e-book* dan *textbook* yang memiliki kajian ilmu terkait *Edge Computing*, *Machine Learning* pada *Artificial Intelligence* dan *Artificial Neural Network.*

II.1.1 Atensi Wajah

Atensi wajah mengacu pada mekanisme atau proses yang memungkinkan sistem untuk fokus pada bagian-bagian tertentu dari citra wajah yang dianggap lebih relevan atau penting. Dalam konteks pengolahan citra atau pengenalan wajah, atensi wajah bertujuan untuk memperkuat atau menyoroti area wajah yang mengandung informasi utama, seperti ekspresi wajah, mata, mulut, atau fitur wajah lainnya, yang dapat digunakan untuk tugas seperti pengakuan wajah, analisis emosi, atau pelacakan wajah (Mujiyanto et al., 2024). Aspek-aspek pada atensi wajah sebagai berikut:

1. Fokus pada Fitur Kritis: Dalam banyak aplikasi, wajah memiliki beberapa fitur yang sangat penting untuk tujuan tertentu, seperti mata dalam pengenalan emosi atau mulut dalam analisis ekspresi wajah. Atensi wajah bertujuan untuk memprioritaskan bagian-bagian ini dalam citra wajah.
2. Peningkatan Akurasi: Dengan memfokuskan pada bagian-bagian yang lebih relevan dari citra wajah, model berbasis atensi dapat meningkatkan akurasi dalam tugas pengolahan citra, seperti pengenalan wajah, deteksi ekspresi emosi, atau bahkan identifikasi individu.
3. Menggunakan Mekanisme Atensi: dalam jaringan saraf dalam (*deep neural networks*) atau jaringan saraf konvolusional (CNN), mekanisme atensi *(attention mechanism)* digunakan untuk memberikan bobot yang

lebih besar pada fitur wajah yang dianggap lebih penting. Hal ini mirip dengan cara manusia mengarahkan perhatian mereka pada fitur yang lebih penting di wajah orang lain saat berbicara atau berinteraksi.

Penerapan atensi wajah (*facial attention*) memiliki beragam aplikasi yang luas dalam bidang pengolahan citra wajah dan pengenalan wajah. Berikut adalah penerapan atensi wajah:

1. Pengenalan Wajah: Dalam sistem pengenalan wajah, atensi wajah membantu meningkatkan akurasi dengan memfokuskan pada bagian wajah yang lebih unik, seperti mata atau bentuk wajah, untuk memverifikasi identitas individu.
2. Analisis Ekspresi Emosi: Pada analisis ekspresi wajah, atensi wajah memfokuskan pada area seperti mulut atau mata untuk mendeteksi emosi tertentu (misalnya, senyum atau marah).
3. Pemantauan Sosial dan Psikologis: Atensi wajah juga digunakan dalam aplikasi yang melibatkan pemantauan perilaku manusia, seperti dalam analisis interaksi sosial atau untuk mendeteksi tanda-tanda perilaku tertentu.

Mekanisme atensi (Ke et al., 2022) dalam pembelajaran mendalam pada dasarnya adalah mekanisme alokasi bobot yang meniru sistem visual dan kognitif manusia. Hal ini memungkinkan jaringan saraf untuk memfokuskan perhatian pada bagian-bagian penting dari data, memungkinkan mereka untuk membuat keputusan dan meningkatkan kinerja serta kemampuan generalisasi model secara keseluruhan. Mekanisme ini digunakan dalam deskripsi gambar untuk menarik perhatian pada aspek-aspek kunci tertentu dari gambar dan menyesuaikan bobot sesuai dengan itu. Dengan menggunakan mekanisme atensi, arsitektur dasar model encoder-decoder dalam tugas deskripsi gambar diperbaiki, memungkinkan decoder untuk secara selektif mendekode fitur posisi dan meningkatkan kinerja model. Mekanisme atensi telah menjadi bagian krusial dalam bidang deskripsi gambar dan masih banyak ruang untuk perbaikan, seperti mengoptimalkan alokasi bobot dan mengurangi redundansi informasi. Contoh mekanisme atensi pada wajah (Xuanhao & Min, 2023) sebagai berikut:

1. Atensi Saluran

   Dalam jaringan saraf konvolusional, mekanisme atensi saluran adalah struktur unik yang mengekstrak informasi lokal dan global dari gambar untuk menangkap kelas fitur tertentu. Nilai bobot antar-saluran digunakan sebagai ukuran pentingnya dalam pemetaan fitur konvolusional, dengan besar bobot yang mewakili derajat korelasi. Hal ini menyelesaikan masalah ketergantungan saluran fitur dengan mempelajari korelasi antar saluran, secara adaptif memberi bobot pada fitur-fitur dari saluran yang berbeda untuk memperkuat fitur yang berguna dan menekan fitur yang tidak berguna, menyaring perhatian spesifik saluran, meningkatkan akurasi seluruh jaringan, dan dengan demikian meningkatkan kinerja model pada tugas-tugas seperti klasifikasi gambar.

2. Atensi Spasial

   Dalam jaringan saraf konvolusional, atensi spasial adalah mekanisme atensi yang krusial. Mekanisme ini dianggap sebagai proses adaptif untuk menyaring wilayah spasial kunci untuk menentukan "di mana harus fokus," yang meningkatkan kemampuan jaringan untuk mengidentifikasi objek-objek penting dalam peta fitur dengan menyaring wilayah-wilayah kunci dan memberikan bobot pada berbagai wilayah peta fitur input dalam dimensi spasial. Hal ini memungkinkan jaringan untuk lebih memfokuskan perhatian pada informasi yang relevan.

II.1.2 *Edge Computing*

Edge Computing merupakan suatu paradigma komputasi yang baru dengan memproses data pada jaringan tepi (F. Liu et al., 2019b). *Edge Computing* memungkinkan sistem untuk melakukan pemrosesan data di tepi jaringan dan di dekat sumber data. Ini mengurangi bandwidth komunikasi yang diperlukan antara sensor dan pusat data dengan melakukan analitik dan pembuatan pengetahuan didekat sumber data. *Kinerja Edge Computing* memungkinan pengembangan untuk (ESF, 2021) sebagai berikut:

1. Mengembangkan dan mengelola aplikasi IoT berbasis komputasi tepi
2. Menghubungkan perangkat IoT dan layanan cloud menggunakan protocol IoT

3. Menyusun Aliran Data secara visual unutk mengelola, menganalisis dan data rute

Komponen kunci dalam komputasi tepi adalah perangkat tepi, yang biasanya disebut dengan server tepi yang terletak lebih dekat dengan ujung jaringan untuk pemrosesan data, komunikasi, dan lainya (Wang et al., 2020). Berdasarkan tuntutan desain yang berbeda, sistem komputasi tepi dapat di klasifikasikan ke dalam 3 kategori yang akan menghasilkan inovasi arsitektur sistem, model pemograman dan berbagai aplikasi.

1. *Push From Cloud*

   Dalam kategori ini, penyedia cloud memberikan layanan dan komputasi ke perangkat tepi untuk meningkatkan lokalitas, mengurangi waktu respons, dan meningkatkan pengalaman pengguna. Sistem perwakilan termasuk Cloudlet, Cachier, AirBox, dan CloudPath. Banyak penyedia layanan komputasi awan tradisional secara aktif mendorong layanan awan lebih dekat dengan pengguna, memperpendek jarak antara pelanggan dan komputasi awan, agar tidak kehilangan pasar ke komputasi tepi seluler (MEC). Misalnya, Microsoft meluncurkan AzureStack pada tahun 2017, yang memungkinkan kemampuan komputasi awan untuk diintegrasikan ke dalam terminal, dan data dapat diproses dan dianalisis pada perangkat terminal.

2. *Pull From IoT*

   Dalam Kategori ini, Aplikasi IoT menarik layanan dan komputasi awan yang jauh ke tepi dekat untuk menangani sejumlah besar data yang dihasilkan oleh perangkat IoT termasuk PCloud, ParaDrop, FocusStack, dan SpanEdge. Kemajuan dalam embedded system-on-a-chip (SoCs) telah memunculkan banyak perangkat IoT yang cukup kuat untuk menjalankan sistem operasi tertanam dan algoritme kompleks. Banyak produsen mengintegrasikan *machine learning* dan bahkan kemampuan pembelajaran mendalam ke dalam perangkat IoT. Memanfaatkan sistem dan alat komputasi tepi, perangkat IoT dapat secara efektif berbagi sumber daya komputasi, penyimpanan, dan jaringan sambil mempertahankan tingkat kemandirian tertentu.

3. *Hybrid Cloud- Edge Analytics*

   Dalam kategori ini, Integrasi keunggulan cloud dan edge memberikan solusi untuk memfasilitasi hasil optimal global dan waktu respons minimum dalam layanan dan aplikasi canggih modern. Sistem perwakilan mencakup sistem komputasi *Firework* dan *Cloud-Sea*. Sistem komputasi tepi tersebut memanfaatkan kekuatan pemrosesan perangkat IoT untuk memfilter, melakukan praproses, dan menggabungkan data IoT sambil menggunakan kekuatan dan fleksibilitas layanan cloud untuk menjalankan analitik kompleks pada data tersebut. Sebagai contoh, Alibaba *Cloud* meluncurkan produk komputasi *edge* IoT pertamanya, LinkEdge, pada tahun 2018, yang memperluas keunggulannya dalam komputasi awan, data besar, dan kecerdasan buatan (AI) ke edge untuk membangun komputasi kolaboratif yang terintegrasi cloud/tepi. sistem; Amazon merilis Amazon Web Services (AWS) Greengrass pada tahun 2017, yang dapat memperluas AWS ke perangkat sehingga perangkat dapat melakukan operasi lokal pada data yang mereka hasilkan, sementara data ditransfer ke cloud untuk pengelolaan, analisis, dan penyimpanan.

Edge computing juga merupakan sebuah proses komputasi yang difokuskan untuk memproses lalu lintas IoT atau *internet of Things* dan dalam edge computing dalam penyimpanan data akan disimpan sedekat mungkin dengan sumber data ke pusat data yang dapat mengurangi latensi serta penggunaan *bandwidth* yang tidak diperlukan. Jadi secara lebih sederhana *edge computing* adalah sebuah proses yang dijalankan seminimal mungkin yang dilakukan dari cloud dan memindahkannya ke tempat lokal. Pada konsep ini di jaringan *edge* dapat membawa komputasi dengan tujuan mengurangi komunikasi jarak jauh antara *client* dan server. *Edge computing* memberikan kecepatan kepada data karena *edge computing* meningkatkan *network speed* dengan menurunkan latency. *Edge computing* terkadang memang tidak bergantung kepada koneksi internet dan edge computing juga hanya mengirimkan data yang relevan ke *cloud*, jadi tidak semua data dapat diambil saat mengalami serangan *cyber*.

Edge computing tidak bergantung dengan internet dan server. Sehingga dengan edge computing pengguna akan memiliki layanan yang tidak akan terganggu. Tidak

hanya itu edge computing dapat menyimpan serta memproses data secara lokal menggunakan microdata centers. *Edge computing* dapat digunakan di IoT yang dapat membuat user mengurangi bandwidth dan kebutuhan penyimpanan data serta menggantikan data center dengan solusi device. Hal ini dapat mengurangi biaya dari mengimplementasi *IoT device* dan *application* secara signifikan (Cao et al., 2020).

Perangkat Keras untuk *Edge/Fog*: Tiga kategori perangkat termasuk dalam tingkat *Edge Computing*: sensor, yang menyediakan data mentah; perangkat pengguna akhir, yang menerima data yang telah diproses; dan server, yang menyediakan daya pemrosesan atau layanan lainnya. Sementara itu, karena data dihasilkan dan digunakan di tingkat yang lebih rendah, konsep jaringan *Edge* hanya mencakup server.

1. *Edge Device*

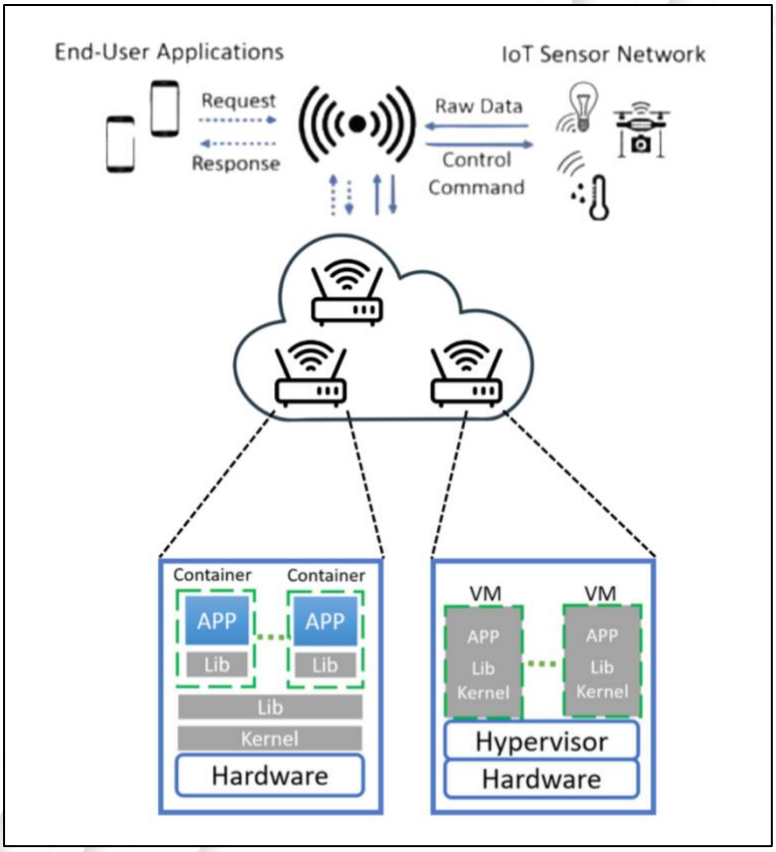
   Semua perangkat yang menghasilkan data dan/atau menjalankan aplikasi pengguna akhir termasuk dalam kategori ini. Perangkat-perangkat ini dapat dibagi menjadi tiga klasifikasi utama: perangkat *mobile*, komputer papan tunggal (*single-board computers)*, dan perangkat terbatas (*restricted devices*). Setiap perangkat IoT yang digunakan untuk berbagai tujuan, termasuk domotika, pengawasan, otomotif, dan banyak lagi, termasuk dalam kategori perangkat terbatas.

2. Server *Edge*

   Server *Edge/Fog:* Terdapat tiga kategori utama perangkat yang dapat berfungsi sebagai server di tingkat *Edge* dan *Fog*: server tujuan umum (yaitu, server yang sama yang digunakan di lingkungan *cloud*), kinerja baru yang dibuat khusus untuk memenuhi kebutuhan *Edge/Fog*, dan kinerja lain yang dibuat untuk kasus penggunaan tertentu (misalnya, automotif).

Virtualisasi di Edge Jaringan: Untuk menawarkan layanan yang lebih skalabel, andal, dan efektif, server Edge dapat memanfaatkan teknologi-teknologi yang mendukung virtualisasi yang telah dibahas pada bagian sebelumnya. Alih-alih mengirimkan data mentah yang terkumpul ke cloud, sensor IoT mengirimkannya ke server Edge/Fog, di mana data tersebut menerima arahan (garis solid). Server-

server ini berada di lokasi geografis yang dekat, namun dapat terhubung langsung ke gateway jaringan IoT (Edge) atau berada di luar jaringan tersebut (Fog). Dengan meluncurkan mesin virtual (VM) atau kontainer yang disesuaikan pada saat permintaan, teknik virtualisasi membantu menjamin skalabilitas jaringan ketika memberikan layanan sesuai permintaan. Jika layanan pengguna akhir tidak dihosting secara lokal, server mengirimkan hasilnya ke cloud setelah mengolah dan menggabungkan informasi yang telah mereka terima. Aplikasi pengguna akhir yang terinstal di smartphone atau perangkat klien lainnya mengurangi latensi dan meningkatkan pengalaman pengguna dengan mengirimkan kueri ke server Edge/Fog (garis putus-putus) daripada ke cloud (Caprolu et al., 2019).



Gambar II.1 Virtualisasi pada Arsitektur Edge

Dalam paradigma komputasi awan (cloud computing), sebagian besar perhitungan terjadi di cloud, yang berarti data dan permintaan diproses di cloud terpusat. Namun, paradigma komputasi seperti ini dapat mengalami latensi yang lebih lama (misalnya, latensi ekor panjang), yang dapat melemahkan pengalaman pengguna. Dalam edge computing, edge memiliki sumber daya komputasi tertentu, yang

memberikan peluang untuk memindahkan sebagian beban kerja dari cloud. Beberapa studi kasus yang pada umumnya menggunakan edge computing sebagai berikut:

1. Cloud Offloading

   Sejumlah penelitian telah membahas tentang cloud offloading dalam hal trade-off energi-kinerja dalam lingkungan mobile-cloud (Yingsaeree et al., 2010). Pada jaringan pengiriman konten tradisional, hanya data yang disimpan di server edge. Ini didasarkan pada kenyataan bahwa penyedia konten menyediakan data di internet, yang benar untuk beberapa dekade terakhir. Dalam IoT, data diproduksi dan dikonsumsi di edge. Oleh karena itu, dalam paradigma edge computing, tidak hanya data tetapi juga operasi yang diterapkan pada data seharusnya disimpan di edge. Salah satu aplikasi potensial yang dapat memperoleh manfaat dari edge computing adalah layanan belanja online. Seorang pelanggan mungkin sering memanipulasi keranjang belanja. Secara default, semua perubahan pada keranjang belanja tersebut akan dilakukan di cloud, dan kemudian tampilan keranjang belanja yang baru akan diperbarui di perangkat pelanggan. Proses ini dapat memakan waktu lama tergantung pada kecepatan jaringan dan tingkat beban server. Proses ini bahkan bisa lebih lama untuk perangkat mobile karena bandwidth jaringan seluler yang relatif rendah. Karena belanja menggunakan perangkat mobile semakin populer, penting untuk meningkatkan pengalaman pengguna, terutama yang terkait dengan latensi. Dalam skenario seperti ini, jika pembaruan keranjang belanja dipindahkan dari server cloud ke node edge, latensi akan berkurang secara dramatis.
2. Video Analytics

   Penyebaran ponsel pintar dan kamera jaringan telah menjadikan analisis video sebagai teknologi yang semakin berkembang. Komputasi awan kini dianggap tidak lagi optimal untuk aplikasi yang memerlukan analisis video, mengingat latensi transmisi data yang tinggi dan masalah terkait privasi. Sebagai ilustrasi, pertimbangkan situasi pencarian anak yang hilang di kota. Saat ini, berbagai jenis kamera banyak dipasang di area perkotaan dan di kendaraan. Ketika seorang anak hilang, kemungkinan besar anak tersebut

akan terekam oleh kamera. Namun, data yang dihasilkan oleh kamera tersebut biasanya tidak diunggah ke cloud karena kekhawatiran tentang privasi atau biaya lalu lintas data, sehingga menyulitkan pemanfaatan data dari kamera yang tersebar luas. Bahkan jika data tersebut tersedia di cloud, proses pengunggahan dan pencarian informasi dalam jumlah besar bisa memakan waktu lama, yang tentu tidak efisien dalam situasi darurat seperti pencarian anak hilang. Dengan paradigma edge computing, permintaan untuk melacak anak yang hilang bisa dibuat dari cloud dan langsung dikirim ke perangkat-perangkat yang ada di area yang relevan. Setiap perangkat, seperti ponsel pintar, dapat memproses permintaan tersebut secara lokal, mencari data pada kamera yang ada di sekitarnya, dan hanya mengirimkan hasilnya kembali ke cloud. Dengan pendekatan ini, data dan kekuatan komputasi yang ada di setiap perangkat dapat dimanfaatkan secara maksimal, memungkinkan pencarian yang jauh lebih cepat dibandingkan dengan mengandalkan komputasi awan terpusat.

3. Smart Home

   IoT dapat memberikan banyak manfaat bagi lingkungan rumah. Beberapa produk telah dikembangkan dan tersedia di pasar, seperti lampu pintar, TV pintar, dan robot vakum. Namun, hanya menambahkan modul Wi-Fi pada perangkat elektronik yang ada dan menghubungkannya ke cloud tidak cukup untuk menciptakan rumah pintar. Dalam lingkungan rumah pintar, selain perangkat yang terhubung, sensor nirkabel murah dan pengendali juga harus dipasang di ruangan, pipa, bahkan lantai dan dinding.

4. *Smart City*

   Paradigma Edge Computing dapat diperluas dengan fleksibel mulai dari satu rumah hingga ke tingkat komunitas, atau bahkan skala kota. *Edge Computing* berpendapat bahwa pemrosesan data harus dilakukan sedekat mungkin dengan sumber data. Dengan desain ini, permintaan dapat dihasilkan dari tingkat atas paradigma komputasi dan diproses langsung di *edge*. *Edge Computing* bisa menjadi kinerja yang ideal untuk kota pintar dengan mempertimbangkan karakteristik-karakteristik (Shi et al., 2016).

5. *Collaborative Edge*

   Cloud computing bisa dibilang telah muncul sebagai kinerja standar untuk memproses sejumlah besar data baik di industri maupun di dunia akademik. Salah satu janji utama dari *cloud computing* adalah bahwa data pada akhirnya akan diproses di *cloud*, baik dengan disimpan di sana atau dipindahkan ke sana. Namun, karena masalah privasi dan biaya transfer data yang tinggi, data milik pemangku kepentingan sangat jarang dibagikan satu sama lain. Akibatnya, peluang untuk kerjasama antar beberapa pemangku kepentingan menjadi sangat terbatas. Konsep logis ini juga bisa mencakup *Edge*, yang merupakan pusat data kecil fisik yang menghubungkan cloud dan pengguna akhir dengan kemampuan pemrosesan data. Meskipun dengan lokasi fisik dan struktur jaringan yang berbeda, *collaborative edge* diusulkan untuk menghubungkan edge dari beberapa pemangku kepentingan yang tersebar secara geografis (Bonomi et al., 2012).

II.1.3 Edge Impulse

Edge Impulse adalah kinerja pengembangan machine learning yang dirancang khusus untuk perangkat *edge*. Kinerja ini memungkinkan pengguna untuk mengumpulkan data, melatih model machine learning, dan mengoptimalkannya untuk perangkat dengan sumber daya terbatas seperti *microcontroller*, sensor, dan perangkat IoT lainnya. *Edge Impulse* menyediakan alat untuk mengumpulkan data dari berbagai sensor dan perangkat *edge*, baik dengan mengunggah data langsung dari perangkat maupun melalui antarmuka pengguna web. Dalam desain dan pelatihan model, kinerja ini menawarkan berbagai algoritma machine learning termasuk klasifikasi, deteksi objek, dan analisis sinyal, yang dapat digunakan untuk melatih model berdasarkan data yang telah dikumpulkan.

Model yang dilatih di *Edge Impulse* dioptimalkan untuk berjalan pada perangkat dengan keterbatasan sumber daya. Optimasi ini meliputi pengurangan ukuran model, efisiensi konsumsi daya, dan peningkatan kecepatan inferensi. Setelah model dilatih dan dioptimalkan, *Edge Impulse* menyediakan cara mudah untuk meng-*deploy* model tersebut ke perangkat edge seperti Arduino, Raspberry Pi, ESP32, dan lainnya. Model dapat diekspor dalam berbagai format yang kompatibel dengan perangkat target.

Selain itu, *Edge Impulse* memiliki fitur dashboard untuk memonitor kinerja model yang di-*deploy*, melihat data real-time, dan mengelola proyek secara keseluruhan. Fitur ini memungkinkan pengguna untuk memantau dan mengatur semua aspek dari proyek *machine learning* mereka dengan lebih efektif (Ziliwu et al., 2024).

II.1.4 *Internet of Things* (IoT)

Pembahsan tentang IoT, meliputi sejarah perkembangan, model referensi, komponen utama dan protokol.

II.1.4.1 Sejarah Perkembangan IoT

Pada abad ke-20, terkenal istilah komputer berotak tanpa indra yang artinya komputer hanya mengetahui apa yang diperintahnya. Hal ini merupakan bentuk keterbatasan karena lebih banyak informasi di dunia ini dari pada yang diketahui oleh manusia dengan mengetik melalui *keyboard* atau *scan barcode*. Tetapi dengan *Internet of Things*, komputer yang dikenal dengan istilah otak tak berindra menjadi komputer yang dapat mengindra sesuatu untuk dirinya (Gabbai, 2015).

Istilah *Internet of Things* pertama kali disebutkan oleh Kevin Ashton pada Tahun 1999 saat presentasi Manajemen *Senior Procter & Gamble*. Kevin Ashton yang saat ini menjadi cofounder and executive director of the Auto-ID Center di *Massachusetts Institute od Technology* (MIT), yang menjelaskan *Radio-frequency identification* (RFID). Dalam presentasinya, Kevin Ashton menggunakan Internet of Things (IoT) untuk menggambarkan sistem sensor bertindak sebagai mata dan telinga dari perangkat komputerisasi (Gabbai, 2015). *Internet of Things* (IoT) merupakan kata kunci hampir di setiap sektor industri, yang menggambarkan era konektifitas masa depan yang diketahui berkembang sangat cepat.

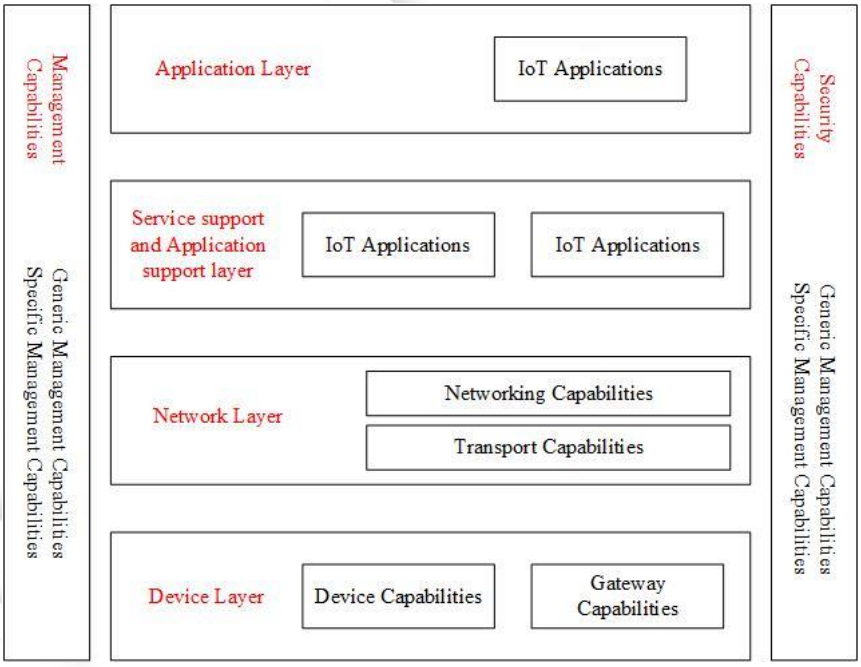
Internet of Things dapat dianggap mampu menyelesaikan permasalahan yang ada dengan implikasi teknologi dan sosial. Dari perspektif standarisasi secara teknik, *Internet of Things* dapat dilihat sebagai infrastruktur global untuk memenuhi kebutuhan informasi masyarakat, yang menyediakan layanan-layanan canggih dengan interkoneksi objek-objek (fisik dan virtual) berdasarkan Teknologi Informasi dan Komunikasi (TIK). Melalui Eksploitasi kemampuan identifikasi, Internet of Things memanfaatkan sepenuhnya objek-objek untuk menyajikan

layanan ke semua jenis aplikasi, memastikan bahwa kebutuhan keamanan dan privasi terpenuhi (ITU-T Y.2060, 2018).

Untuk memahami definisi dari Internet of Things dapat dilihat dari gabungan dari 2 kata yakni ”*Internet*” dan ”*Things*”. ”Internet” didefinisikan sebagai sebuah jaringan komputer yang menggunakan protokol- protokol internet (TCP/IP) yang digunakan untuk berkomunikasi dan berbagi informasi dalam lingkup tertentu. Sementara ”*Things*” dapat diartikan sebagai objek-objek dari dunia fisik yang diambil melalui sensor-sensor yangkemudian dikirim melalui Internet. Namun, dari hasil objek yang telah dikirimkan masih memerlukan penyajian ulang yang diharapkan dapat lebihmudah dimengerti oleh stackholder. Untuk mempermudah model penyimpanan dan pertukaran informasi diperlukan adanya Teknologi *Semantic*. Oleh karena itu untuk mewujudkan *Internet of Things* diperlukan 3 komponen pendukung yakni *Internet, Things* dan *Semantic* (Atzori et al., 2010).

II.1.4.2 Model Referensi IoT

Model Referensi IoT terdiri dari kapabilitas manajemen dan keamanan yang terkait dengan 4 (empat) layer yaitu *Application Layer*, *Service Support and Application Support Layer*, *Network Layer*, and *Device Layer*.



Gambar II.2 Model Referensi IoT

1. *Application Layer*

   Application Layer berisi tentang Aplikasi *Internet of Things* (IoT)

2. *Service Support* and *Application Support Layer*

   Pada layer ini dikelompokan menjadi 2 kapabilitas sebagai berikut

   a) Kapabilitas Dukungan Generik merupakan Kapabilitas umum yang digunakan oleh aplikasi IoT yang berbeda, seperti pengolahan data atau penyimpanan data

   b) Kapabilitas Dukungan Khusus merupakan Kapabilitas tertentu yang memenuhi kebutuhan aplikasi yang beragam, yang dapat terdiiri dari berbagai kelompok kapabilitas yang lebih khusus dalam rangka memberikan fungsi pendukung yang berbeda untuk aplikasi IoT yang berbeda.

3. *Network Layer*

   Pada layar ini dibagi menjadi 2 (dua) bagian sebagai berikut

   a) Kapabilitas Jaringan menyediakan fungsi kontrol yang relevan, seperti akses, fungsi kontrol transportasi, manajemen mobilitas, otentikasi, otorisasi dan akuntansi.

   b) Kapabilitas Transportasi menyediakan konektivitas untuk pengangkutan layanan IoT dan aplikasi informasi data khusus serta transportasi kontrol IoT dan manajemen informasi

4. *Device Layer*

   Pada *device layer*, dapat diklasifikasikan menjadi 2 (dua) bagian sebagai berikut

   a) Kapabilitas Perangkat termasuk namun tidak terbatas pada interaksi langsung dengan jaringan komunikasi, interaksi tidak langsung dengan jaringan komunikasi, jaringan Ad-Hoc, Tidur dan Bangun.

   b) Kapabilitas Gateway termasuk namun tidak terbatas pada dukungan beberapa *interface*, dan konversi protokol.

5. *Management Capabilities*

   Kapabilitas Manajemen IoT meliputi kesalahan tradisional, konfigurasi, *accounting*, kinerjansi dan keamanan, contohnya seperti kesalahan

manajemen, konfigurasi manajemen, akuntansi manajemen, kinerjansi manajemen dan keamanan manajemen.

Kapabilitas manajemen IoT dapat dikategorikan ke dalam kapabilitas manajemen generik dan kapabilitas manajemen tertentu. Kapabilitas manajemen generik esensial dalam IoT meliputi:

a) Manajemen Perangkat: seperti diantaranya aktivasi perangkat remot dan de-aktivasi, diagnostik, firmware atau memperbarui perangkat lunak, manajemen status dari perangkat kerja
b) Manajemen Topologi Jaringan Lokal
c) Manajemen *Traffic dan Congestion*: seperti mengatasi jaringan yang sedang overload. Serta kapablitas manajemen tertentu yang digabungkan erat dengan kebutuhan aplikasi spesifik, seperti kebutuhan pemantauan daya saluran transmisi dari *smart grid.*

6. *Security Capabilities*

Ada 2 (dua) jenis kapabilitas keamanan yakni kapabilitas generik dan kapabilitas keamanan tertentu. Kapabilitas keamanan generik dari aplikasi termasuk kedalam penjelasan berikut:

a) Pada *Application Layer*: otorisasi, otentikasi, aplikasi kerahasiaan data dan perlindungan integritas, perlindungan privasi, audit keamanan dan anti-virus.
b) Pada *Network Layer*: otorisasi, otentikasi, penggunaan data dan data yang menandakan kerahasiaan serta sinyal perlindungan intergritas.
c) Pada *Device Layer*: otentikasi, otorisasi, perangkat validasi integritas, kontrol akses, kerahasiaan data dan perlindungan integritas.

Kapabilitas keamanan juga dapat digabungkan dengan kebutuhan aplikasi-aplikasi tertentu, seperti mobile payment dan kebutuhan keamanan (ITU-T Y.2060, 2018).

II.1.4.3 Komponen Utama Sistem IoT

Internet of Things mencakup berbagai macam teknologi inovatif dengan modul yang terintegrasi dalam komponen perangkat keras, memungkinkan solusi cerdas.

1. Perangkat Keras pada IoT

   Perangkat keras yang digunakan dalam sistem *Internet of Things* berkisar dari perangkat untuk kontrol, papan dasbor, router hingga server dan sensor. Perangkat ini menangani tugas berulang seperti deteksi ancaman, aktivasi sistem, pemeriksaan keamanan, dan tindakan khusus dukungan. Blok bangunan perangkat keras IoT adalah sebagai berikut:

   a) Aset yang akan dikontrol atau dipantau
   b) Modul akuisisi data
   c) Modul pemrosesan data
   d) Modul komunikasi

   Aset yang akan dikontrol atau dimonitor dapat berdiri sendiri, atau tergabung dalam perangkat pintar. Modul akuisisi data adalah salah satu yang berfokus pada perolehan sinyal dari dunia luar dan mengubahnya menjadi sinyal digital. Ini adalah blok perangkat keras yang berisi semua sensor, sehingga membantu memperoleh sinyal dunia nyata, seperti cahaya, suhu, getaran, atau tekanan.

   Komponen perangkat keras ini juga mencakup konversi sinyal sensor menjadi informasi digital. Modul pemrosesan data adalah blok bangunan ketiga dari perangkat IoT. Ini mewakili unit yang memproses data dan melakukan operasi di atasnya, serta menyimpannya. Ini juga membutuhkan kemampuan penyimpanan. Ini terjadi karena beberapa perangkat data memproses sendiri data yang diperoleh, alih-alih mengirimkannya ke hulu. Di sisi lain, terdapat perangkat IoT yang tidak memiliki kemampuan tersebut dan membutuhkan entitas perantara untuk menyimpan dan mengolah data. Entitas ini adalah perangkat gate-way atau aplikasi cloud yang digunakan untuk agregasi lebih lanjut. Modul komunikasi adalah bagian yang memungkinkan komunikasi dengan komponen pihak ketiga dan kinerja khusus cloud.

2. Kerangka Kerja dan Komponen IoT

   Dari perspektif lain, kerangka kerja IoT dapat dibagi menjadi empat komponen utama, yang terdiri dari layer fisik dan virtual. Setiap segmen kerangka melakukan peran yang terdefinisi dengan baik dalam proses menghubungkan dunia fisik dan digital. Layer ini adalah: *end nodes*, *network layer*, *service layer*, dan application (BIRLOG et al., 2020)

a) *End nodes*

Node akhir sistem IoT hanya berisi komponen perangkat keras yang sebagian besar direpresentasikan oleh sensor dan aktuator. Seperti namanya, sensor terdiri dari modul penginderaan, beserta modul daya, energi, frekuensi radio, dan manajemen. Komunikasi dilakukan menggunakan modul frekuensi radio, melalui fitur pemrosesan sinyalnya. Sensor mendapatkan namanya dari modul terpentingnya, yaitu modul penginderaan. Ini digunakan untuk man-age sensing menggunakan beberapa perangkat pengukuran aktif atau pasif, seperti: *accelerome-ters*, magnetometer, sensor gambar, sensor cahaya, sensor kelembaban, sensor tekanan, sensor akustik, giroskop, dan sensor proximity. Aktuator adalah komponen dari layer fisik yang dirancang untuk menjadi mitra sen-sors. Perangkat ini mengintervensi realitas fisik, saat menerima sinyal kendali. Jika mereka memiliki sumber energi, mereka akan mengubahnya menjadi gerakan mekanis, sesuai dengan sinyal yang diterima.

b) *Network layer*

Teknologi yang digunakan dalam hal ini termasuk Internet, jaringan komunikasi bergerak, jaringan sensor nirkabel, infrastruktur jaringan, dan protokol komunikasi. Jaringan komunikasi bergerak merupakan jenis jaringan yang terdiri dari kumpulan terminal yang saling berhubungan, node dan orang yang menggunakan perangkat seluler portabel. Jaringan sensor nirkabel adalah sekelompok sensor yang tersebar secara spasial yang digunakan untuk memantau kondisi lingkungan, seperti tekanan, suhu, gerakan, atau kelembaban. Biasanya jaringan sensor nirkabel berisi ratusan ribu node sensor. Node sensor dapat berkomunikasi di antara mereka sendiri menggunakan sinyal radio. Node sensor nirkabel dilengkapi dengan perangkat penginderaan dan komputasi, pemancar gelombang radio, dan komponen daya. Protokol komunikasi IoT adalah komponen perangkat lunak dari layer jaringan. Mereka memiliki kemampuan berbeda berdasarkan kompleksitas solusi IoT yang diimplementasikan. Protokol yang paling terkenal adalah: *Bluetooth Low Energy, Wifi*, *Near Field Communications*, LoRaWAN, Z-Wave dan ZigBee.

c) *Service layer*

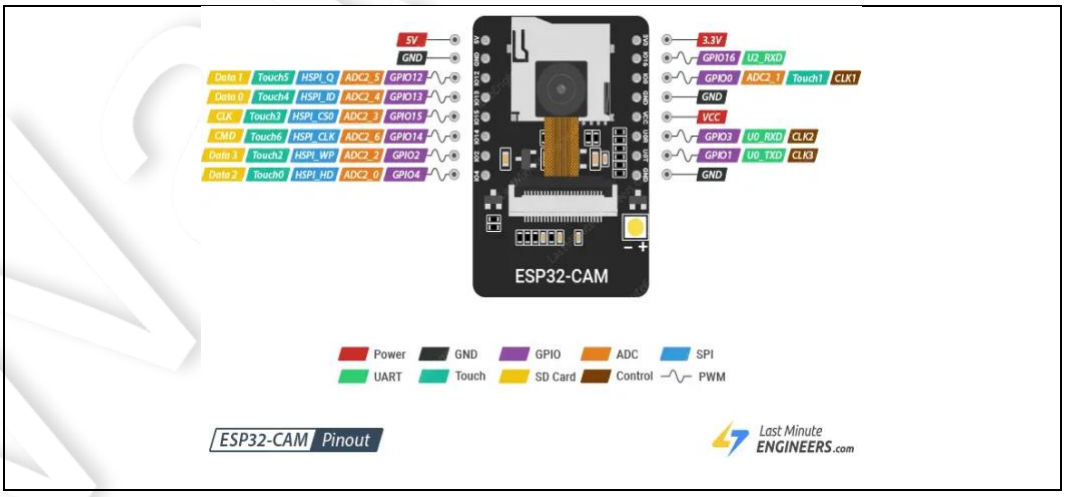
Service layer yang mempresentasikan entitas penghubung antar network layer bawah dan application layer atas. Komponen ini memperkenalkan gagasan *cloud* dan *fog service*, karena telah disediakan computational power yang diperlukan. Pada layer ini serangan siber menargetkan informasi pribadi dan rahasia, perangkat akhir IoT, fungsi monitor dan kontrol.

d) *Application layer*

Menyediakan mekanisme yang memungkinkan integrasi layanan yang termasuk dalam layer bawah melalui teknologi seluler. Layer ini bertanggungjawab atas pembagian data disepanjang jaringan IoT oleh karena itu perlu memastikan keamanam, privasi data, kontrol akses, dan untuk mencegah kebocoran informasi.

II.1.4.4 ESP32CAM

Papan pengembangan yang menggunakan modul kamera berbasis ESP32. ESP32 CAM dilengkapi dengan kamera, Bluetooth, WiFi, dan slot kartu microSD. Dibandingkan dengan perangkat ESP sebelumnya, ESP32 CAM memiliki lebih sedikit pin I/O, dengan hanya 10 pin GPIO yang dapat diakses. Pin lainnya pada ESP32 CAM digunakan secara internal oleh kamera dan slot kartu microSD. Karena modul ESP32 CAM tidak memiliki port microUSB, diperlukan adaptor FTDI tambahan. Adaptor FTDI digunakan untuk menghubungkan modul ESP32 CAM ke komputer. Komponen utama dari modul ESP32 CAM adalah kameranya. Sensor kamera yang terdapat pada modul ESP32 CAM adalah OV2640. Sensor chip OV2640 dapat digunakan untuk pengenalan objek dan wajah.



Spesifikasi ESP32 CAM antara lain:	
1) 802.11b/g/n Wi-Fi 2) Bluetooth 4.2 with BLE 3) Kamera 2MP 4) UART, SPI, I2C and PWM interfaces 5) Clock speed up to 160 MHz 6) Computing power up to 600 DMIPS	7) In-Built Micro SD Card slot 8) Image transfer rate 15-60 frame per second 9) 520 KB SRAM dan 4 MB PSRAM 10) Upload gambar WiFi 11) Built-in Flash LED

Terdapat beberapa Pin dalam ESP32CAM. Pin merujuk pada fisik atau port input/output digital yang dapat digunakan untuk berbagai fungsi, seperti menghubungkan sensor, aktuator, atau komponen lainnya. Setiap pin pada ESP32-CAM memiliki nomor yang unik dan dapat dikonfigurasi untuk berbagai keperluan melalui perangkat lunak. Pin yang terdapat pada ESP32 CAM:

a) Serial/UART pin

Pin Name	Function
GPIO1	UOTXD / TX (Transmission Pin)
GPIO3	UOTXD / TX (Reception Pin)

b) Flash Mode Pin

Pin GPIO0 dengan menghubungkan dengan ground memungkinkan ESP32 CAM akan berada pada flashing mode untuk menggunggah code program.

a) SD Card Connector Pins

b) Serial/UART pin

Pin Name	SD Card Connection
GPIO2	Data0 pin (RTC & ADC supported)
GPIO4	Data1 pin (RTC & ADC supported)
GPIO12	Data2 pin (RTC & ADC supported)
GPIO13	Data3 pin (RTC & ADC supported)
GPIO14	CLK (RTC & ADC supported)
GPIO15	CMD (RTC & ADC supported)

II.1.4.5 Arduino IDE

Arduino IDE (*Integrated Development Environment)* adalah perangkat lunak open source yang digunakan untuk membuat dan mengunggah program ke board Arduino. Perangkat lunak ini dapat diunduh secara gratis dari situs resmi Arduino. Arduino IDE dikembangkan menggunakan bahasa pemrograman Java dan dilengkapi dengan library C/C++. Arduino IDE memungkinkan pengguna untuk mengedit, membuat, memvalidasi kode, dan mengunggah program ke board Arduino. Kode program yang dibuat di Arduino disebut "*sketch*" atau *source code* Arduino, dan file tersebut disimpan dengan ekstensi .ino.

Arduino IDE menyediakan library, yaitu kumpulan kode yang membantu dalam memberikan perintah kepada komponen agar berfungsi sesuai dengan yang diinginkan. Library ini memudahkan pengembang dalam menulis sketch atau program. Terdapat dua jenis library di Arduino: library bawaan dan library tambahan. Library bawaan sudah ter-install di Arduino IDE dan dapat langsung digunakan dalam program, sedangkan library tambahan perlu diunduh dari internet untuk mendukung jalannya program. Untuk melihat library yang tersedia di Arduino, pengguna dapat memilih menu Sketch, kemudian Include Library (Slamet Purwo Santoso & Fajar Wijayanto, 2022).

II.1.4.6 Raspberry Pi

Raspberry Pi adalah sebuah single-board computer yang berukuran kecil, sehingga mudah untuk digunakan sebagai otak dari projek-projek Internet of Things. Raspberry Pi sendiri dapat digunakan selayaknya komputer pada umumnya, karena telah memiliki prosesor berarsitektur ARM (*Advanced RSIC Machine*) yang dikemas dan diintegrasikan diatas PCB (*Printed Circuit Board*). Untuk penyimpanan data serta Operating Systemnya, Raspberry Pi menggunakan memori external berupa SD card sebanyak satu buah dengan kapasitas penyimpanan minimal 4GB untuk pengoperasian sistemnya.

Seperti yang telah dijelaskan, Raspberry Pi memiliki komponen-komponen yang hampir sama dengan komputer pada umumnya, seperti contohnya Raspberry Pi memiliki CPU, Integrated GPU, beberapa port USB, dll. Untuk sumber daya nya sendiri, Raspberry Pi mengandalkan micro-USB power dengan daya minimal

sebesar 5V dan minimal arus 700 mA. Selain memiliki komponen-komponen yang hampir sama dengan komputer pada umumnya, Raspberry Pi juga dapat digunakan selayaknya komputer pada umumnya juga. Misalnya, untuk mengetik dokumen, sebagai pemutar musik atau pemutar video, dan masih banyak lagi. Namun, banyak pula yang memanfaatkan Raspberry Pi ini untuk sebuah microcontroller untuk sistem buatan mereka sendiri, yang biasanya terintegrasi dengan internet. Sebagai contoh, untuk pengatur suhu ruangan otomatis, sistem *image or object character recognition,* dll. Hal ini dapat terjadi dikarenakan karakter Raspberry Pi yang modular, Raspberry Pi dapat diintegrasikan dengan berbagai modul dan juga kamera Raspberry Pi dengan OS (*Operating System*) raspbian atau OS (*Operating System*) yang berbasis Linux lainnya mendukung beberapa pemrograman, yang paling populer adalah PERL dan python.

II.1.5 *Artificial Neural Network*

Perceptron adalah jenis *Artificial Neural Network* (ANN) yang paling sederhana, yaitu jaringan neural dengan satu lapisan yang terdiri dari nilai input, bobot, bias, jumlah net, dan fungsi aktivasi. Perceptron biasanya digunakan sebagai pengklasifikasi biner dalam pembelajaran terawasi (*supervised learning*) untuk mengkategorikan input ke dalam salah satu dari dua kelas. Bobot mewakili pentingnya setiap input, dan jumlah bobot yang terakumulasi dari input digunakan oleh fungsi aktivasi untuk mengklasifikasikan data.

Namun, perceptron hanya dapat digunakan untuk masalah biner yang sederhana. Oleh karena itu, banyak jenis jaringan neural lain yang dikembangkan dengan mengambil perceptron sebagai elemen dasar. Pengembangan ini biasanya melibatkan penggunaan beberapa lapisan perceptron, dengan satu atau lebih perceptron di setiap lapisan. Perceptron-perceptron, yang juga disebut sebagai node atau neuron, di setiap lapisan membuat kesimpulan berdasarkan hasil yang diperoleh dari lapisan sebelumnya dan kemudian mengirimkan hasil tersebut ke perceptron pada lapisan berikutnya. Terdapat tiga jenis utama *Artificial Neural Networks* (ANNs), yaitu *Feedforward Neural Networks* (FNNs), *Recurrent Neural Networks* (RNNs), dan *Convolutional Neural Networks* (CNNs) (Emambocus et al., 2023).

II.1.5.1 Feedforward Neural Networks (FNN)

Feedforward Neural Network (FNN), yang juga sering disebut sebagai *Fully Connected Neural Network* (FC atau FCNN), adalah salah satu model jaringan syaraf yang paling awal dan paling sederhana. Jaringan ini terdiri dari beberapa lapisan node komputasi yang sepenuhnya terhubung dan diorganisasikan dalam banyak lapisan. Nilai dari setiap node di lapisan tersembunyi atau output dihitung dengan mengambil jumlah bobot dari semua node pada lapisan sebelumnya, kemudian melewatkan nilai tersebut ke fungsi nonlinier seperti sigmoid, tanh, dan ReLU.

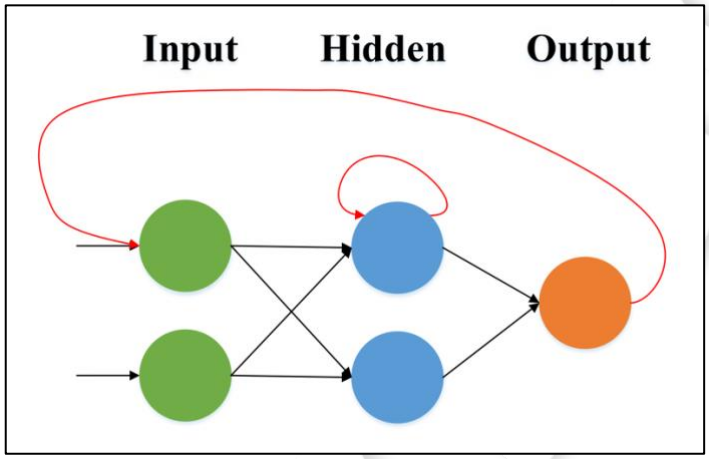
Struktur FNN yang sepenuhnya terhubung memungkinkan setiap lapisan untuk memproses kombinasi dari semua fitur lapisan sebelumnya. Namun, hal ini juga menjadi kelemahan karena koneksi penuh tersebut menghasilkan jumlah parameter yang sangat besar. Akibatnya, proses pelatihan FNN bisa memakan waktu yang cukup lama. Selain itu, FNN tidak memiliki kemampuan untuk secara eksplisit menangkap data spasial atau temporal.

Untuk prediksi aliran lalu lintas, FNN biasanya berperan sebagai bagian dari jaringan dalam bentuk hybrid deep network, yang tujuan utamanya adalah untuk melakukan tugas-tugas seperti menggabungkan output dari berbagai komponen dalam jaringan, transformasi dimensi, dan memasukkan data eksternal seperti informasi cuaca. Hal ini karena ukuran lapisan input atau output dapat diatur secara manual, yang memberi kemampuan pada FNN untuk mengubah input dengan dimensi sembarang menjadi output dengan dimensi sembarang. Ketika digunakan untuk mengintegrasi data eksternal, input bergantung pada jenis data eksternal yang digunakan. Nilai numerik dapat disediakan begitu saja, sementara nilai kategorikal perlu diubah terlebih dahulu (misalnya, dengan menggunakan one-hot encoding). Untuk penggabungan output dan transformasi dimensi, input sepenuhnya bergantung pada model yang digunakan (Tedjopurnomo et al., 2022).

II.1.5.2 *Recurrent Neural Networks* (RNN)

Pada *Artificial Neural Network* (ANN) tradisional, diasumsikan bahwa semua input atau semua output adalah independen satu sama lain. Namun, untuk banyak tugas, input (atau output) saling terkait. Sebagai contoh, untuk memprediksi pola

mobilitas perangkat nirkabel, data input yaitu lokasi pengguna tentu saja saling berhubungan. Untuk itu, *Recurrent Neural Networks* (RNNs), yang merupakan arsitektur ANN yang memungkinkan koneksi neuron dari neuron di satu lapisan ke neuron-neuron di lapisan sebelumnya, seperti yang ditunjukkan pada Gambar II.3, telah diperkenalkan. Perubahan yang tampaknya sederhana ini memungkinkan output dari jaringan syaraf tidak hanya bergantung pada input saat ini, tetapi juga pada input historis.



Gambar II.3 Arsitektur *Recurrent Neural Networks* (RNN)

Hal ini memungkinkan RNN untuk memanfaatkan informasi sekuensial dan mengeksploitasi perilaku temporal dinamis, seperti yang dihadapi dalam prediksi mobilitas atau pengenalan suara. Sebagai contoh, RNN dapat digunakan untuk memprediksi pola mobilitas perangkat seluler dan pengguna nirkabel. Pola-pola ini berkaitan dengan lokasi historis yang pernah dikunjungi oleh pengguna nirkabel. Tugas ini tidak dapat dilakukan dalam satu langkah tanpa menggabungkan lokasi-lokasi historis dari langkah-langkah sebelumnya. Oleh karena itu, ANNs yang output-nya hanya bergantung pada input saat ini, seperti FNNs, tidak dapat melakukan tugas-tugas yang sangat bergantung pada waktu seperti ini (Chen et al., 2019).

II.1.5.3 Convolution Neural Network (CNN)

Machine Learning (ML) adalah bidang studi yang berfokus pada desain dan analisis algoritma yang memungkinkan komputer untuk belajar dari data. Menurut Samuel, ML melibatkan algoritma generik yang dapat menghasilkan keluaran yang berguna dari sejumlah data tanpa memerlukan penulisan kode spesifik. Inti dari algoritma

generik ini adalah kemampuannya untuk membuat aturan atau model dari data yang diberikan. Machine learning juga dapat diartikan sebuah komputer yang memiliki kemampuan belajar tanpa diprogram secara eksplisit. Program tersebut memanfaatkan data untuk membangun model dan mengambil keputusan berdasarkan model yang telah dibangun. Kemampuan komputasi merupakan peralatan yang dibutuhkan untuk mengoperasikan ML. Jika di analogikan, dasar-dasar matematika sebagai otak dan kemampuan komputasi sebagai otot dan badannya yang menggerakkan. Kedua hal tersebut saling melengkapi.

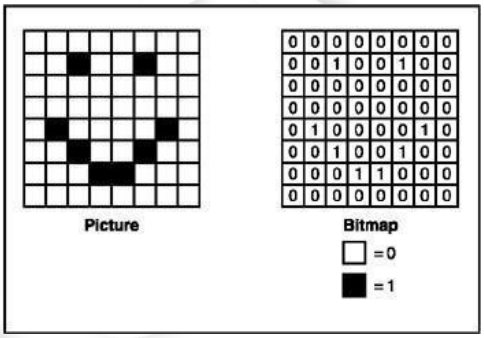
Kemampuan komputasi identik dengan kemampuan untuk menggunakan bahasa pemrograman atau tool berbasis komputer tertentu untuk memproses data. Saat ini banyak aplikasi atau tool yang dapat digunakan untuk menyelesaikan permasalahan ML atau pembelajaran mesin. Sederhananya, pengguna tinggal memasukkan data lalu aplikasi tersebut akan memproses data lalu menghasilkan model yang dapat digunakan untuk melakukan prediksi. Sebagian besar aplikasi tersebut sudah mengimplementasi algoritma-algoritma yang populer yang dapat menyelesaikan masalah (Ibnu Daqiqil, 2021).

Baru-baru ini jaringan saraf terutama *Convolution Neural Network* (CNN) telah mencapai keberhasilan yang signifikan di banyak bidang termausk pemaham gambar/video, pemrosesan dan kompresi dan lainnya. Sebuah CNN bias any aterdiri dari satu atau lebih lapisan konvolusi. Secara khususm beberapa tugas juga menambahkan beberapa lapisan yang terhubung penuh setelah lapisan konvolusi. Parameter di lapisan ini dapat dilatih dengan baik berdasarkan gambar dan video besar yang diberi label untuk tugas tertentu dalam strategi ujung ke ujung. CNN terlatih dapat diterapkan dengan baik untuk menyelesaikan tugas klasifikasi, pengenalan dan prediksi pada data uji dengan kemampuan beradaptasi yang sangat efesien. Kualitas sinyal prediksi yang dihasilkan oleh CNN telah melampaui predictor berbasis aturan. Selain itu CNN dapat diartikan sebagai pengekstrak fitur untuk mengubah gambar dan video menjadi ruang fitur dengan representasi yang ringkas, yang bermanfaat untuk kompresi gambar dan video.

Berdasarkan karakteristiknya, CNN telah diakui sebagai solusi untuk kompresi. CNN mengungguli algoritma tradisional dengan margin besar dalam tugas-tugas

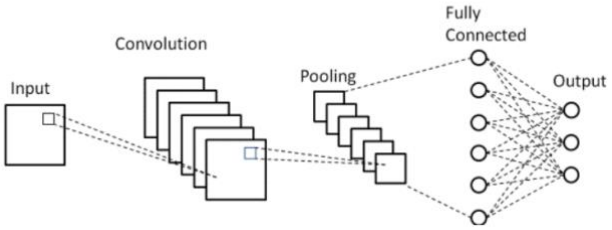
visi computer tingkat tinggi seperti klasifikasi gambar dan deteksi objek. Untuk banyak tugas visi komputer tingkat rendah, CNN mencapai kinerja yang sangat mengesankan, misalnya, resolusi super dan pengurangan artefak kompresi. CNN mengadopsi operasi konvolusi untuk mengkarakterisasi korelasi antara piksel dan operasi konvolusi berjenjang sesuai dengan sifat statistik hierarkis dari gambar alami. Selain itu, bidang reseptif lokal dan bobot yang diperkenalkan oleh operasi konvolusi juga menurunkan parameter CNN yang dapat dilatih, yang secara signifikan mengurangi risiko masalah over-fitting (Ma et al., 2020).

Convolutional Neural Network (CNN) merupakan salah satu model deep learning yang banyak digunakan untuk keperluan analisis citra. Secara prinsip, CNN meniru cara kerja otak manusia untuk mengenali objek yang dilihatnya. Dengan bantuan CNN, kini komputer dapat "melihat" dan "membedakan berbagai objek". Fitur ini disebut "image recognition". Tentu saja mata manusia berbeda dengan "mata" komputer. Mata manusia melihat objek gambar seperti biasa, Sedangkan komputer melihat gambar di atas sebagai array dari angka-angka yang merupakan nilai piksel, seperti pada Gambar II.4.



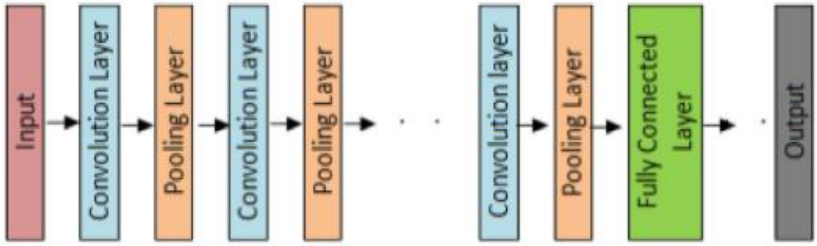
Gambar II.4 Nilai Pixel

Bentuk array sangat bergantung pada resolusi dan ukuran gambar. Secara garis besar, CNN tidak jauh berbeda dengan neural network biasa. CNN terdiri dari neuron-neuron yang memiliki weight, bias, dan activation function. Umumnya, CNN terdiri dari beberapa lapisan yang masing-masing memiliki peran khusus dalam proses analisis Gambar II.5.



Gambar II.5 Layer CNN

Model CNN yang mendalam terdiri dari sejumlah *processing layers* yang terbatas yang dapat mempelajari berbagai fitur dari data input (misalnya gambar) dengan berbagai tingkat abstraksi. Lapisan inisiasi mempelajari dan mengekstrak fitur tingkat tinggi (dengan abstraksi lebih rendah), sementara lapisan ini lebih dalam mempelajari dan mengekstrak fitur tingkat rendah (denfan abstraksi lebih tinggi). Model konseptual dasar CNN ditunjukkan pada Gambar II.6, dengan berbagai jenis lapisan.



Gambar II.6 Layer MobileNetV2

MobileNetV2 adalah sebuah arsitektur jaringan saraf tiruan yang dirancang khusus untuk pengolahan gambar pada perangkat seluler dengan sumber daya terbatas seperti CPU, RAM, dan daya baterai. Pada Proses ekstraksi fitur menggunakan MobileNetV2 yang telah dilatih sebelumnya oleh ImageNet sebagai ekstraktor fitur. MobileNetV2 yang telah dilatih sebelumnya pada kumpulan data ImageNet yang merupakan kumpulan data besar dari 1,4 juta gambar dan 1000 kelas gambar web, digunakan untuk mengekstrak fitur citra iris mata dalam kumpulan dataset citra iris mata. Selanjutnya untuk proses klasifikasi menggunakan lapisan konvolusi 1x1 Global Average Polling dan lapisan Dense dengan aktivasi Sigmoid.

Arsitektur MobileNetV2 menggunakan teknik *Depthwise Separable Convolution* untuk mengurangi kompleksitas komputasi dengan membagi konvolusi menjadi dua tahap: *Depthwise Convolution* dan *Pointwise Convolution.* Pada tahap *Depthwise Convolution*, filter diterapkan pada setiap saluran (channel) gambar dan menghasilkan keluaran setara dengan konvolusi standar, namun lebih efisien karena mengurangi jumlah parameter yang digunakan dalam proses tersebut. Sedangkan Pointwise Convolution bertujuan untuk menggabungkan hasil dari *Depthwise Convolution*. Selain teknik *Depthwise Separable Convolution*, MobileNetV2 juga memanfaatkan teknik linear bottlenecks. Linear bottlenecks digunakan untuk melakukan reduksi dimensi pada lapisan yang memiliki dimensi besar. Reduksi dimensi ini dilakukan dengan menggunakan konvolusi 1x1 untuk mengurangi jumlah saluran pada lapisan tersebut. Kemudian, ekspansi dimensi dilakukan dengan menggunakan konvolusi 1x1 dan 3x3 untuk mengembalikan jumlah saluran pada lapisan tersebut (Zhao et al., 2024).

II.2 Tinjauan Penelitian Terkait

Dalam penelitian ini, dilakukan tinjauan literatur terhadap penelitian - penelitian sebelumnya yang memiliki relevan dengan penelitian ini. Hal ini dilakukan untuk mengidentifikasi perbedaan yang ada antara penelitian ini dan penelitian - penelitian terdahulu yang relevan. Secara umum pengelompokan literatur yang dijadikan referensi utama yang berkaitan dengan latar bekang dalam penelitian ini sebagai berikut

Tabel II.1 Tinjauan Pustaka Penelitian

No	Peneliti	Data	Metode	Hasil
1	Bo Jiang, et.al (2019)	Head pose	Convolutional Neural Network (CNN)	Akurasi head detection sebesar 97,4% dan akurasi head pose sebesar 94,6%.
2	Sadia Afroze et.al (2022)	9 pose kepala	Multi-task Cascade Neural Network (MTCNN) dan ResNet18.	Hasil eksperimen menunjukkan bahwa model yang diusulkan mengungguli model deep learning lainnya dengan mencapai akurasi tertinggi pada tiga dataset: BIWI-M (96,97%), Pointing'04-M

No	Peneliti	Data	Metode	Hasil
				(96,04%), dan HPoD 9 (98,99%). Model visual focus of attention mendapatkan akurasi 94,12% pada skenario multi-objek.
3	Partha Chakraborty et.al (2020)	Data lingkungan dan data eyeball	Logistic Regression, Support Vector Machine (SVM), Decision Tree, K-Nearest Neighbor (KNN), AdaBoost, Multilayer Perceptron (MLP), Extra Tree Classifier, dan Voting Classifier	Logistic Regression mencapai akurasi tertinggi sebesar 96%, mengungguli yang lain dalam memprediksi kelas-kelas ini. Voting Classifier berbobot gabungan mencapai akurasi 95%.
4	Liu, T, et.al (2021)	gambar pose kepala inframerah seperti dataset IRHPE (primer), CAS-PEAL-R1, dan pointing 04 (sekunder)	Penentuan label gambar pose menggunakan Gaussian nonuniform, neural network	Hasil penelitian menunjukkan bahwa NGDNet secara signifikan mengungguli algoritma konvensional dalam hal estimasi pose kepala inframerah. Model NGDNet mencapai akurasi 77,39% pada dataset IRHPE, 99,08% pada dataset CAS-PEAL-R1, dan 87,41% pada dataset Pointing'04
5	Tripti Singh, et.al (2021)	300W-LP (300W across large poses) dataset, AFLW2000 (Annotated Facial Landmarks in the Wild) dataset, AFLW2000 (Annotated Facial Landmarks in the Wild) dataset.	regresi elastic net, CPAM, DNN	model yang diusulkan dapat menangani dataset besar, pemrosesan data waktu nyata, variasi pose, oklusi parsial, dan ekspresi wajah yang beragam dengan kesalahan absolut rata-rata (MAE) sebesar 3° atau kurang. Evaluasi pada tiga dataset standar (300W-LP, AFLW2000, dan NIMH-ChEFS) menunjukkan bahwa kinerja model ini sebanding dengan metodologi terbaru seperti AADL, JFA, dan RAFA-Net, mencapai MAE hingga 6°.

No	Peneliti	Data	Metode	Hasil
6	Sadia Afroze, et.al (2020)	data train menggunakan 7 jam vide dengan 435.000 frame dan 9 pose kepala. Data uji terdiri dari 81.000 frame	MTCNN, IoU dan P-Net, YOLO	Model klasifikasi mencapai akurasi 97,00% pada set uji. dan dalam skenario multi-objek, akurasi mendekati 95,00%.
7	Chongyang Bai, et.al (2019)	Dataset baru terdiri dari 5 video (35 orang, 109 menit, 7604 label) dari permainan Resistance.	Model ICAF (Iterative Collective Attention Focus) untuk klasifikasi kolektif. Setiap orang dimodelkan menggunakan pengklasifikasi terpisah. Prediksi dari pengklasifikasi orang lain digunakan sebagai input untuk setiap pengklasifikasi individu, menggabungkan ketergantungan perilaku.	ICAF menunjukkan peningkatan akurasi sebesar 1%-5% dibandingkan baseline terkuat dalam memprediksi fokus perhatian visual.
8	Patricia Goldberg, et.al (2019)	data video rekaman kelas yang dianotasi secara manual (mengkode perilaku siswa setiap detik menggunakan software CARMA dengan skala -2 hingga +2) dan Metode otomatis menggunakan pendekatan vision-based machine learning dengan fitur seperti pose kepala, arah pandangan, dan ekspresi wajah	Algoritma yang digunakan adalah Support Vector Regression (SVR) untuk mengestimasi intensitas keterlibatan siswa.	Hasil menunjukkan bahwa kombinasi pose kepala, arah pandangan, dan ekspresi wajah memiliki korelasi yang tinggi dengan anotasi manual (r = 0.61). Akurasi prediksi keterlibatan yang dihitung dengan mean squared error (MSE) dan koefisien korelasi Pearson bervariasi, dengan model terbaik yang menggabungkan sinkroni antar siswa menunjukkan peningkatan 9% dalam korelasi dengan anotasi manual.

No	Peneliti	Data	Metode	Hasil
9	P. Chakraborty et al (2021)	Laporan survei (data lingkungan) dan data bola mata (pergerakan bola mata, waktu membaca, dan putaran kepala)	Regresi Logistik, SVM, Decision Tree, KNN, AdaBoost, MLP, Extra Tree Classifier, Voting Classifier	Regresi Logistik: Akurasi 96%, Voting Classifier: Akurasi 95%
10	Benoit Massé, Silèye Ba, dan Radu Horaud (2017)	Dua dataset yang tersedia secara publik yang mengandung interaksi manusia-robot dan manusia-manusia multi-partai	Model ruang keadaan switching Bayesian dengan variabel laten untuk VFOA dan tatapan mata	Prosedur pembelajaran yang dapat ditangani dan algoritma efisien untuk melacak tatapan dan fokus visual
11	Vaisnavi C dan Suja Palaniswamy (2022)	Dataset CAFE (Children Affective Facial Expression) yang mencakup tujuh emosi dasar (Neutral, Angry, Surprise, Sad, Fear, Disgust)	Meta-learning, Layer-based CNN, SNSER (Siamese Network for Student Emotion Recognition Model)	Akurasi 80% dalam mengenali emosi
12	Haipeng Zeng, et.al (2021)	Video kelas yang dikumpulkan dari berbagai taman kanak-kanak. Setiap video berdurasi 10 menit dengan resolusi 1920 x 1080 dan 30 fps. Total frame per video adalah 18.000.	Face detection menggunakan algoritma MTCNN, face recognition menggunakan Facenet untuk vektorisasi gambar dan pengenalan wajah, serta emotion recognition menggunakan model CNN (ResNet-50) yang mengkategorikan ekspresi ke dalam tujuh emosi.	Akurasi deteksi wajah sebesar 95,9% pada video taman kanak-kanak dan 96,3% pada video seminar. Akurasi pengenalan emosi mencapai 64,8% pada video taman kanak-kanak dan 68,5% pada video seminar.
13	Vidia Kamath, et,al (2024)	Dataset MOD-2022 disiapkan untuk deteksi dan pelacakan objek, mencakup berbagai kelas	Model VGG16-SSD dilatih menggunakan dataset ini	Model VGG16-SSD yang dilatih dengan pendekatan ini menunjukkan penurunan kinerja pada gambar berkualitas rendah, tetapi masih dapat

No	Peneliti	Data	Metode	Hasil
		dan bebas dari kesalahan pelabelan		mendeteksi dan melacak objek dengan akurasi yang dapat diterima pada perangkat terbatas seperti Raspberry Pi.
14	Paul D. Rosero-Montalvo, et.al (2024)	Penelitian ini menggunakan lima dataset yang berbeda untuk tugas deteksi dan klasifikasi, dengan total 122.124 gambar.	Metode optimisasi arsitektur CNN dengan TensorFlow Lite (Tf-lite) dan kuantisasi (IQ) untuk perangkat edge	Model Tf-lite menggunakan semua inti CPU di bawah 80%, dengan VGG-16 mencapai 100% penggunaan inti, sedangkan model kuantisasi menggunakan satu inti CPU lebih dari 90% dan dua inti di bawah 20%. Ini menunjukkan perbedaan dalam penggunaan inti CPU dan konsumsi daya.
15	Neha Patil, et.al (2017)	Gambar gerakan atau gestur yang diambil menggunakan webcam standar.	Metode yang digunakan adalah Raspberry Pi dengan OpenCV untuk pemrosesan gambar dan deteksi gerakan atau gestur.	Mmenunjukkan sistem mampu untuk memantau dan memberikan peringatan saat gerakan atau gestur terdeteksi, dan mengirim ke server cloud atau menyimpannya secara lokal, dengan mempertimbangkan efisiensi konsumsi daya.
16	Zehra Ozkan. et.al (2021)	Penelitian ini menggunakan data berupa 3948 gambar yang diambil dari berbagai kondisi cuaca, lingkungan, dan waktu untuk melatih model deteksi dan pengenalan ancaman	Metode yang digunakan adalah pembelajaran mesin dengan jaringan saraf konvolusional (CNN), di mana dua arsitektur, Faster-RCNN dan SSD MobileNet V2, dipilih untuk membandingkan akurasi deteksi.	Hasil menunjukkan bahwa deteksi dan pengenalan objek dengan Faster-RCNN mencapai akurasi 91%, sementara SSD MobileNet V2 mencapai akurasi 88%.
17	Annisa Firasanti. et.al (2022)	Deteksi penggunaan masker di pintu pintar	TensorFlow dan OpenCV pada Raspberry Pi	Akurasi deteksi masker sebesar 98,11% pada jarak efektif 2,7 m.

No	Peneliti	Data	Metode	Hasil
18	Mohd Nadhir, et.al (2021)	Ekspresi wajah (FER-2013 dataset)	CNN untuk ekstraksi fitur, KNN untuk pengenalan ekspresi	Akurasi FER sebesar 75,26% pada Raspberry Pi dengan keterbatasan sumber daya komputasi dan memori.
19	Mokhamad Arfan Wicaksono dan Yoanes Bandung (2022)	Data multimedia dari beberapa node sensor	Evaluasi protokol CoAP pada jaringan sensor multimedia nirkabel	Penurunan throughput rata-rata secara linier seiring dengan bertambahnya perangkat.
20	Leonardo Gianto. et.al (2023)	Suhu tubuh dan gambar wajah	Iot 3-lapis dengan sensor, edge computing, cloud computing	Sistem berhasil mendeteksi individu dengan penyakit infeksius demam di area publik dalam ruangan.

Dalam literatur yang diatas, integrasi antara machine learning dan edge computing umumnya mengacu pada pemrosesan data secara lokal pada perangkat edge (seperti perangkat IoT atau mikrokontroler) untuk mengurangi latensi dan ketergantungan pada cloud. Pendekatan ini banyak digunakan dalam aplikasi yang membutuhkan respons cepat dan pemrosesan data dalam waktu nyata, seperti dalam sistem pengenalan wajah, deteksi objek, dan pemantauan kesehatan.

Contoh integrasi yang sudah ada adalah sistem yang memanfaatkan edge devices seperti Raspberry Pi atau NVIDIA Jetson yang menjalankan model machine learning secara lokal. Model-model ini dilatih menggunakan dataset besar di cloud atau server, lalu dipindahkan dan dijalankan di perangkat edge untuk analisis data secara langsung. Misalnya, sistem yang menggabungkan TensorFlow Lite atau OpenVINO untuk mempercepat inferensi model di perangkat edge. Pendekatan ini mengurangi kebutuhan akan *bandwidth* dan latensi jaringan, namun tetap mengandalkan perangkat keras yang lebih besar dan lebih mahal.

Alternarif baru yang dikembangkan pada penelitian ini dalam integrasi *machine learning* dan *edge computing*, dengan memanfaatkan perangkat ESP32-CAM, yang merupakan modul kamera terjangkau dengan kemampuan Wi-Fi dan pemrosesan data di *edge*. Berbeda dengan sistem yang menggunakan perangkat edge yang lebih besar dan lebih mahal, penelitian ini menggunakan ESP32-CAM untuk menangkap

gambar secara real-time dan memprosesnya menggunakan model machine learning berbasis MobileNetV2 yang telah dikembangkan dan dioptimalkan menggunakan Edge Impulse. Keunikan dari pendekatan ini adalah:

1. Penggunaan ESP32-CAM sebagai perangkat edge kecil dan terjangkau, yang memungkinkan implementasi sistem pengolahan gambar *real-time* dengan biaya yang jauh lebih rendah dibandingkan dengan perangkat seperti Raspberry Pi atau Jetson.
2. Integrasi model machine learning secara langsung di perangkat *edge* tanpa memerlukan koneksi ke cloud untuk inferensi, sehingga mengurangi ketergantungan pada koneksi internet dan meningkatkan efisiensi dalam hal waktu respons dan penggunaan bandwidth.
3. Pemanfaatan *Edge Impulse* untuk pengembangan model *machine learning* yang diunggah dan dijalankan langsung di ESP32-CAM, memungkinkan pemrosesan gambar fokus/tidak fokus secara otomatis dan real-time.

Dengan demikian, alternatif yang diusulkan dalam penelitian ini memberikan solusi yang lebih hemat biaya, lebih efisien, dan lebih mudah diimplementasikan dalam berbagai aplikasi yang membutuhkan pengolahan data gambar secara real-time, seperti dalam sistem pengawasan atau perangkat IoT berbasis kamera.

Bab III Metodologi Penelitian

Pada proses penelitian salah satu hal yang terpenting adalah pengunaan metodologi pada penelitian, dikarenakan metodologi penelitian itu adalah cara atau strategi yang digunakan oleh peneliti dalam mengumpulkan data penelitian, dimulai dari mencari informasi/data yang dibutuhkan untuk memecahkan berbagai masalah yang bertujuan memberikan solusi atas masalah tersebut.

III.1 Metode Penelitian

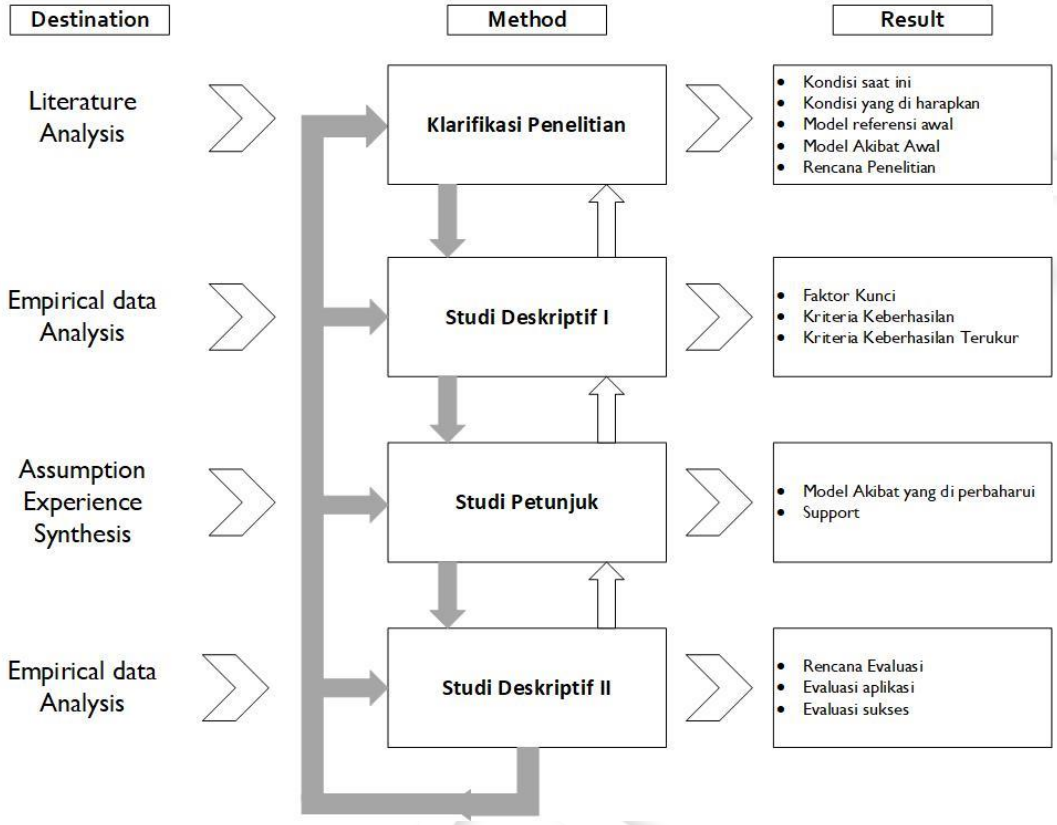
Desain penelitian yang dilakukan dalam penelitian ini adalah *Design Research Methodology* (DRM). *Design Research Methodology* (DRM) adalah pendekatan yang menggambarkan serangkaian metode dan pedoman pendukung yang dapat digunakan sebagai kerangka kerja atau desain awal untuk melakukan peneltian.

Design Research Methodology (DRM) untuk mendukung pendekatan penelitian yang lebih ketat dengan membantu merencanakan dan mengimplementasikan penelitian desain. Metodologi yang digunakan secara fleksibel, akan membantu membuat penelitian desain lebih efektif dan efisien. Tujuan khusus DRM (Blessing & Chakrabarti, 2009) adalah sebagai berikut

1. Untuk membantu mengidentifikasi bidang penelitian, proyek dan program yang paling mungkin bermanfaat secara akademis dan praktis serta realistis
2. Untuk memungkinkan berbagai pendekatan dan metode penelitian
3. Memberikan pedoman untuk perencanaan penelitian yang sistematis
4. Memberikan pedoman untuk penelitian yang lebih ketat
5. Untuk membantu mengembangkan garis argumentasi yang kuat
6. Memberikan metode dan petunjuk baru pada metode yang ada untuk melaksanakan tahapan proses penelitian
7. Untuk membantu memilih metode dan kombinasi metode yang sesuai
8. Untuk memberikan konteks untuk memposisikan proyek dan program penelitian relatif terhadap penelitian desain lainnya
9. Untuk mendorong refleksi atas pendekatan yang diterapkan.

Design Research Methodology (DRM) terdiri dari empat tahap: Research *Clarification, Descriptive Study I* (DS-I), *Prescriptive Study* (PS) *and Descriptive*

Study II (DS-II). Pada Gambar III.1 menunjukkan hubungan antara tahap-tahap ini, sarana dasar yang digunakan di setiap tahap, dan hasil utama.



Gambar III.1 Tahapan *Design Research Methodology* (DRM) (Blessing & Chakrabarti, 2009)

III.1.1 Research Clarification

Keterkaitan *Research Clarification* sebagai bagian dari metode, dikaitkan dengan tujuan dan hasil dapat dilihat pada Tabel III.1

Tabel III.1 Keterkaitan Research Clarification dengan Tujuan dan Hasil

Tujuan (*destination*)	Metode (*method*)	Hasil (*result*)
1. Literature Analysis 2. Pengelompokan tema literatur, meliputi: *a. Machine Learning* *b. Edge Computing*	1. **Tujuan**: mengimplementasikan machine learning untuk deteksi kefokusan audiens berdasarkan wajah pada peralatan edge computing berbasis Internet of Thing dalam Kegiatan Belajar Mengajar (KBM). 2. Masalah: Bagaimana membangun sistem deteksi fokus dan tidak fokus pada image wajah berbasis edge computing dan machine learning dengan menggunakan	1. Kondisi Saat ini: Belum ada edge computing untuk klasifikasi fokus dan tidak fokus berdasarkan image wajah yang tercapture oleh *Edge Device* ESP32-CAM 2. Kondisi yang diharapkan: Adanya machine learning berbasis edge computing

Tujuan (*destination*)	Metode (*method*)	Hasil (*result*)
	Convolutional Neural Networks (CNN) dengan Model Arsitektur MobileNet V2 yang diterapkan menggunakan ESP32-CAM dan Raspberry pi, guna mendukung peningkatan kualitas Kegiatan Belajar Mengajar 3. Literatur utama diantaranya; (a) Machine Learning (b) Convolutional Neural Networks (CNN) (c) Edge Computing (d) Internet of Thing (e) Server 4. Research question yang telah terjawab berdasarkan literatur; (a) Bagaimana melakukan training model (b) Bagaimana melakukan testing pada model (c) Bagaiamana mengklasifikasi menggunakan *Convolutional Neural Networks* (CNN) (d) Bagaimana Komunikasi *Edge Device* dengan *Storage Device* 5. Tipe *design research* adalah [Research Clarification({Review-based}] →[Descriptive Study I{Review-based}] →[Prescriptive Study{Review-based},{Intial/Comprehensive}] →[Descriptive Study II{Comprehensive})] →[Prescriptive Study{Review-based},{Intial/Comprehensive}] 6. Komunikasi data dan Komputasi, Komunikasi data: ESP32-CAM ke Server Komputasi: Machine Learning dengan Edge Computing di ESP32-CAM 7. Tahapan penelitian secara keseluruhan dan global, yaitu: a) Mendesain model machine learning b) Menentukan dataset training c) Mengklasifikasi data uji dengan *Convolutional Neural Networks* (CNN) di Edge Device ESP32-CAM d) Menyimpan hasil klasifikasi di server melalui jaringan IoT	yang mampu mengklasifikasi fokus dan tidak fokus 3. Model referrensi awal: Machine learning berbasis *Convolutional Neural Networks* (CNN) berupa Arsitektur MobileNet V2 4. Model akibat awal: Komputasi machine learning bisa dilakukan pada *Edge Device* ESP32-CAM 5. Rencana Penelitian: a) Mendesain model *machine learning* b) Menentukan dataset training c) Intruksi-intruksi model *machine learning* dapat disimpan di *Edge Device* d) ESP32-CAM dapat mengcapture *image* e) Mengklasifikasi data uji dengan *Convolutional Neural Networks* (CNN) di Edge Device ESP32-CAM f) *Edge Device* ESP32-CAM dapat mengirim data ke Server g) Menyimpan hasil klasifikasi di server melalui jaringan IoT

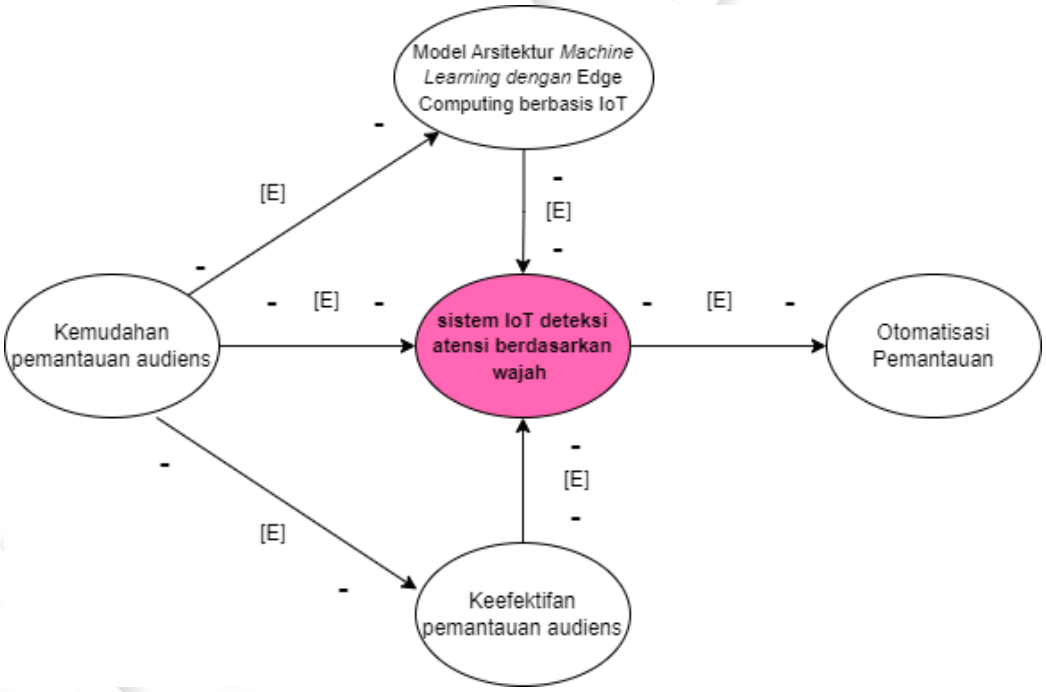
Hasil dari tahap klarifikasi penelitian, membuat asusmsi awal yang didukung dengan pencarian fakta-fakta untuk merumuskan tujuan penelitian. Pada tahap ini melakukan analisis kebutuhan berdasarkan analisis literatur yang menghasilkan

kondisi saat ini, kondisi yang diharapkan, model referensi awal dan model akibat awal. Berikut Tabel III.2 menunjukan kondisi saat ini dan kondisi yang diharapkan.

Tabel III.2 Kondisi saat ini dan kondisi yang diharapkan

Kondisi saat ini	Kondisi yang diharapkan
1. Model *machine learning* pada umumnya dijalankan pada PC/Server 2. Belum ada sistem pendeteksi fokus dan tidak fokus secara otomatis berdasarkan image wajah pada saat kegiatan belajar mengajar baik daring maupun luring 3. Belum ada *edge computing* untuk klasifikasi fokus dan tidak fokus berdasarkan image wajah yang tercapture oleh *Edge Device* ESP32-CAM	1. Model *machine learning* dapat dijalankan pada *Edge Device* untuk mendukung *Edge Computing* berbasis kecerdasan buatan 2. Terimplementasi sistem pendeteksi fokus dan tidak fokus secara otomatis berdasarkan image wajah pada saat kegiatan belajar mengajar baik daring maupun luring 3. Adanya *machine learning* berbasis *edge computing* yang mampu mengklasifikasi fokus dan tidak fokus

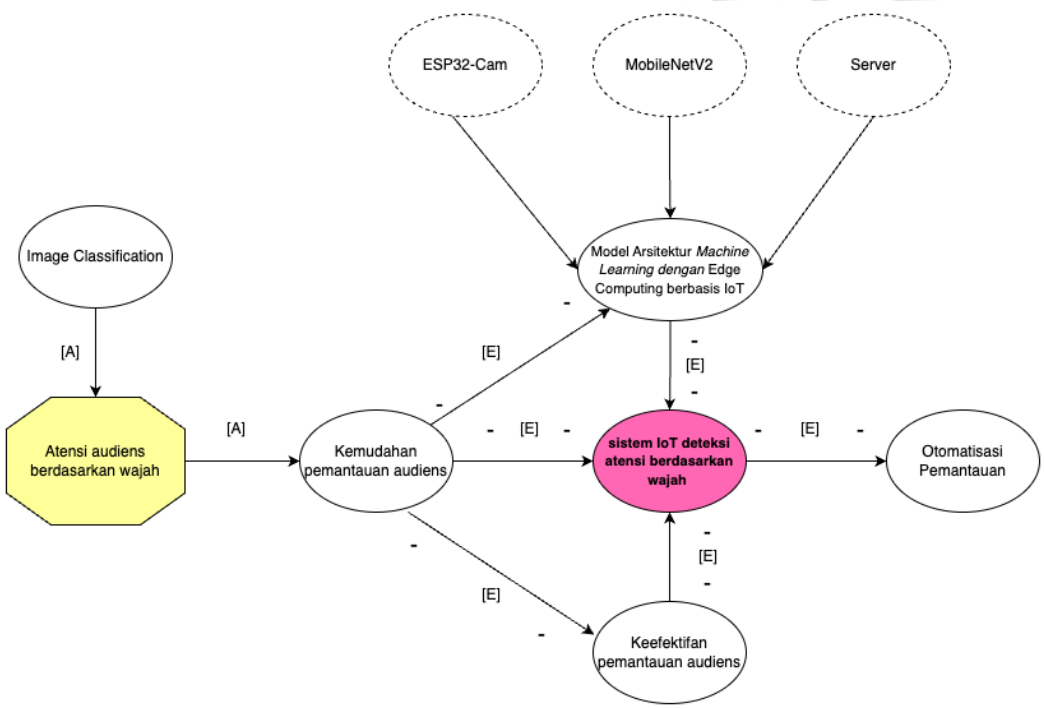
Berdasarkan Tabel III.2 yang menjabarkan hasil analisis kondisi saat ini dapat dirancang model referensi awal yang menjadi dasar penelitian. Berikut model referensi awal yang diilustrasikan pada Gambar III.2 sebagai berikut.



Gambar III.2 Model Referensi Awal

Model referensi awal menggambarkan perlu otomatisasi pemantauan atensi audiens berdasarkan citra wajah dalam kegiatan belajar mengajar dengan memanfaatkan teknologi *edge computing* untuk *machine learning* yang berada dalam jaringan IoT. Pemantauan yang efektif meruapakan kriteria awal yang menjadi target solusi dari penelitian ini dan akan dirancang dukungan (support) agar dapat meningkatkan target solusi.

Berdasarkan Tabel III.2 yang menjabarkan hasil analisis kondisi yang diharapkan dapat dirancang model referensi awal yang menjadi dasar penelitian. Berikut model akinat awal yang diilustrasikan pada Gambar III.3 sebagai berikut.



Gambar III.3 Model Akibat Awal

Model akibat awal menggambarkan model yang ditambahkan dengan dukungan (*support*) dalam kotak segidelapan. Support tersebut digunakan sebagai dukungan dalam mendeteksi atensi fokus dan tidak fokus berdasarkan citra wajah yang telah diklasifikasi melibatkan kecerdasan buatan.

III.1.2 Descriptive Study I (DS-I)

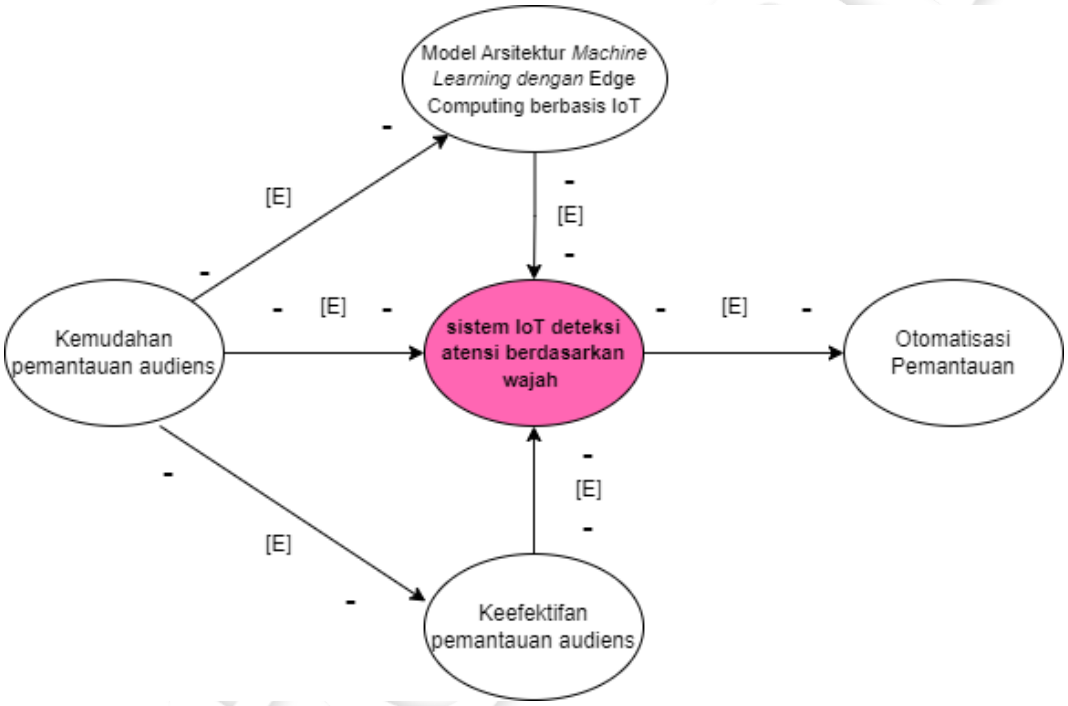
Keterkaitan *Descriptive Study* I (DS-I) sebagai bagian dari metode, dikaitkan dengan tujuan dan hasil dapat dilihat pada Tabel III.3

Tabel III.3 Keterkaitan *Descriptive Study* I dengan Tujuan dan Hasil

Tujuan (*destination*)	Metode (*method*)	Hasil (*result*)
Empirical data Analysis: Data Image format file *.JPG/JPEG	1. Literatur yang terkait diantaranya: a. *Machine Learning*: (Li et al., 2022) b. *Edge Computing*: (Jovovic et al., 2022) 2. Fokus peneltian diantaranya: a. Deteksi atensi fokus dan tidak fokus dengan *machine learning* b. Model machine learning pada edge computing c. Komunikasi data Edge Device pada jaringan IoT. 3. Developing research plan for DS-I a. Data: Image format file *.JPG/JPEG b. *Method*: Model Training, Model Testing, Klasifikasi data uji, Pengiriman data dan Pengiriman informasi *c. Hardware*: ESP32-CAM, Raspberry Pi d. *Software*: *Edge Impulse, Arduino could* 4. *Undertaking empirical study*. a. *Collecting data*: Data image diperoleh dari Edge Device ESP32-CAM dan dikirim ke Server b. *Processing data*: Klasifikasi image wajah fokus dan tidak fokus c. *Analysing and interpreting data*: Model data training dan model data testing di Edge Impulse d. *Verifying the result*: Pelabelan image fokus dan tidak fokus dapat dilihat menggunakan Server Raspberry pi e. *Drawing Conclusions*: (a) Men-training (b) Capture image (c) Klasifikasi image (d) Mengirim dan menyimpan di server *5. Drawing overall conclusions* a) Mendesain model machine learning b) Menentukan dataset training	1. Faktor Kunci: Model *machine learning* MobileNetV2 dapat dijalankan di *edge device* ESP32-CAM 2. Kriteria Keberhasilan: Mengklasifikasi image wajah ke dalam 2 kelas (fokus dan tidak fokus) bergantung pada model *machine learning* yang ada di ESP32-CAM 3. Kriteria Keberhasilan Terukur: Tingkat akurasi hasil klasifikasi pada proses *edge computing machine learning*

Tujuan (*destination*)	Metode (*method*)	Hasil (*result*)
	c) Mengklasifikasi data uji dengan Convolutional Neural Networks (CNN) di Edge Device ESP32-CAM d) Menyimpan hasil klasifikasi di server melalui jaringan IoT	

Hasil dari tahap tahap Studi Deskriptif I membuat model referensi berdasarkan model referensi awal dan model akibat awal yang telah disusun sebelumnya. Berikut Gambar III.4 menunjukkan model referensi yang sudah ditentukan *key factor, success criteria,* dan *measurable criteria*.



Gambar III.4 Model Referensi

Model referensi ini menggambarkan pemahaman mengenai situasi yang mempengaruhi kriteria awal dengan menentukan *key factor, success criteria,* dan *measurable criteria.* Pada penelitian ini yang menjadi *key factor* adalah Model machine learning MobileNetV2 dapat dijalankan di edge device ESP32-CAM. *Measurable criteria* digunakan untuk mengukur kesuksesan tingkat akurasi model data training. Pada penelitian ini, yang menjadi *measurable criteria* adalah Tingkat akurasi hasil klasifikasi pada proses edge computing machine learning *Success criteria* merupakan tujuan akhir dari kontribusi penelitian ini dan sebagai penentuan tingkat kesuksesan model. *Success criteria* pada penelitian ini adalah

mengklasifikasi image wajah ke dalam 2 kelas (fokus dan tidak fokus) bergantung pada model *machine learning* yang ada di ESP32-CAM.

III.1.3 Prescriptive Study (PS)

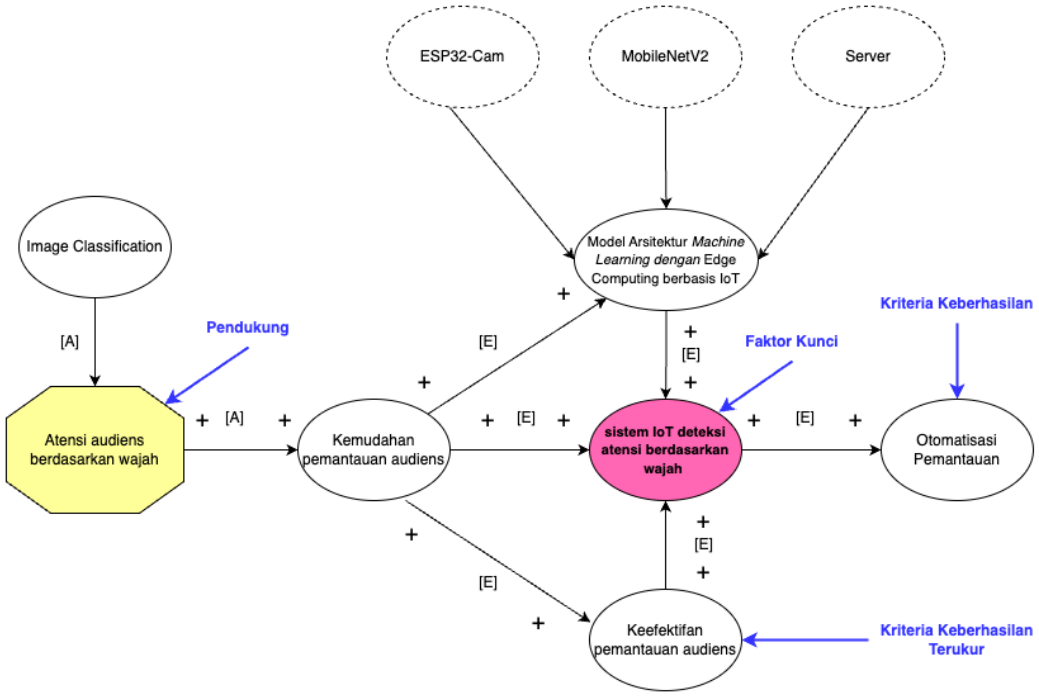
Keterkaitan *Prescriptive Study* (PS) sebagai bagian dari metode, dikaitkan dengan tujuan dan hasil dapat dilihat pada Tabel III.4

Tabel III.4 Keterkaitan *Prescriptive Study* (PS) dengan Tujuan dan Hasil

Tujuan (d*estination*)	Metode (*method*)	Hasil (*result*)
Assumption Experience Synthesis: Menjalankan komputasi model *machine learning* arsitektur MobileNetV2 di *edge device* ESP32-CAM.	1. Kriteria sukses terukur, diantaranya a. Model *machine learning* dapat memproses dataset sebagai data training. Tujuannya mendapatkan tingkat akurasi yang tinggi pada saat testing model arsitektur MobileNetV2. b. Model *machine learning* arsitektur MobileNetV2 dapat mengklasifikasi *image* data test yang berasal dari image capture ESP32-CAM berupa wajah audiens dikategorikan menjadi 2 kelompok yaitu fokus dan tidak fokus dengan tingkat akurasi tinggi. 2. Function dan Sub-function yaitu: a) Mendesain model *machine learning* b) Menentukan dataset training c) Intruksi-intruksi model *machine learning* dapat disimpan di *Edge Device* d) ESP32-CAM dapat mengcapture *image* e) Mengklasifikasi data uji dengan *Convolutional Neural Networks* (CNN) di Edge Device ESP32-CAM f) *Edge Device* ESP32-CAM dapat mengirim data ke server g) Menyimpan hasil klasifikasi di server melalui jaringan IoT 3. Elaborasi *a)* Intended Support: Edge Impulse (membuat model training dataset), server (untuk menyimpan dan menampilkan hasil klasifikasi), MobileNetV2 (untuk mengklasifikasi *image* wajah fokus dan tidak fokus).	1. Model akibat yang diperbaharui: Mengotomatisasi pemantauan atensi audiens yang efektif 2. Support: Impulse (membuat model training dataset), server (untuk menyimpan dan menampilkan hasil klasifikasi), MobileNetV2 (untuk mengklasifikasi image wajah fokus dan tidak fokus).

Tujuan (*destination*)	Metode (*method*)	Hasil (*result*)
	b) Intended Introduction Plan: MobileNetV2 dirancang untuk memiliki ukuran yang kecil dan efisiensi yang tinggi dalam penggunaan sumber daya komputasi sehingga model machine learning ini cocok digunakan pada edge computing dengan jaringan IoT. c) Intended Impact Model: Tingkat akurasi yang tinggi pada model saat training dataset, dan pada saat testing model arsitektur MobileNetV2 4. Realisasi *Core function* pada *Intended Support* dan *Outline evaluation plan*, diantaranya: a) Mendesain model machine learning b) Menentukan dataset training c) Mengklasifikasi data uji dengan *Convolutional Neural Networks* (CNN) di Edge Device ESP32-CAM d) Menyimpan hasil klasifikasi di server melalui jaringan IoT e) Mengotomatisasi pemantauan atensi audiens yang efektif *5. Support Evaluation* *Actual Support* yang diuji berdasarkan literatur diantaranya: a) Pengumpulan dataset b) Training model c) Testing klasifikasi data tes d) Running model pada *edge device* e) Simpan hasil klasifikasi pada *server*	

Hasil dari tahap Studi Preskriptif ini, menyusun model akibat dengan memperbaiki asumsi awal sehingga memperjelas permasalahan penelitian yang dikaji. Gambar III.5 menunjukkan model akibat yang disusun.



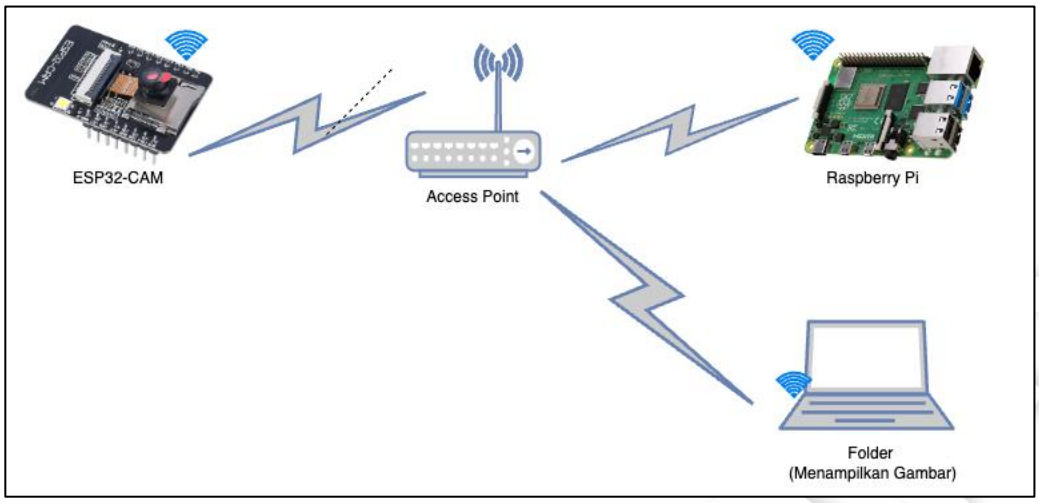
Gambar III.5 Model Akibat

III.1.4 Descriptive Study II (DS-II)

Pada tahap Studi Deskriptif II ini, melakukan evaluasi dampak dari penerapan support yang dikembangkan. Keluaran dari tahap ini adalah hasil dan analisis penelitian dan usulan untuk penelitian selanjutnya.

III.2 Desain Penelitian

Penelitian ini ditunjukan untuk memantau secara otomatis atensi audiens berdasarkan image wajah dalam kegiatan belajar mengajar, sehingga dapat diketahui ketidakfokusan audiens. Adapun tahapan utama dalam penelitian ini adalah sebegai berikut, (a) mendesain model machine learning, (b) Menentukan dataset training (c) Mengklasifikasi data uji dengan *Convolutional Neural Networks* (CNN) di ESP32-CAM (d) Menyimpan hasil klasifikasi di server melalui jaringan IoT.



Gambar III.6 Desain Penelitian

Pada Gambar III.6 ditunjukan tentang desain dari penelitian ini yaitu (a) ESP32-CAM menangkap gambar, mengklasifikasikan gambar, dan mengirim data (b) Meneruskan data ke server (c) Menyimpan data di server (d) menampilkan gambar dan label yang disimpan.

III.3 Metode Pengumpulan Data

Metode pengumpulan data pada penelitian ini sebagai berikut

A. Data

Dataset untuk melakukan training model berbasis *Convolutional Neural Networks* (CNN) dengan arsitektur MobileNetV2 berasal dari koleksi hasil *capture* peralatan ESP32-CAM dengan berbagai pose wajah dan dilabeli dengan nama label fokus dan tidak fokus dengan menggunakan bantuan aplikasi Edge Impluse.

B. Teknik Pengumpulan Data

ESP32-CAM mengcapture image wajah audiens dalam format *.JPG, kemudian diklasifikasi berdasarkan model yang ada sebagai data testing dan image wajah yang telah dilabeli disimpan pada server.

III.4 Metode Analisis Data

Terdapat proses analisis data yang dilakukan penelitian ini meliputi;

1. ***Resolusi image***, pada proses mencapture data image.

2. ***File size***, pada saat image akan dijadikan sebagai data training dan image dijadikan sebagai data testing.
3. **Keakuratan**, pada training model dengan menggunakan dataset dan testing model pada saat mengklasifikasi image wajah inputan dari edge device ESP32-CAM.
4. **Sukses pengiriman**, pada saat pentransfer data image dari ESP32-CAM langsung diproses pada edge device tersebut dimasukan kedalam model, kemudian hasil proses dari model dikirim ke server.

Bab IV Analisis Model

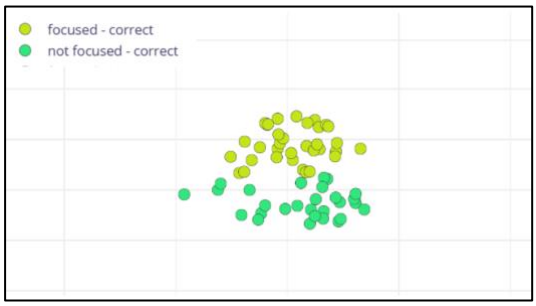
Penelitian ini menggunakan metode CRISP-DM (*Cross-Industry Standard Process for Data Mining*), yang merupakan pendekatan terstruktur untuk merencanakan, melaksanakan, dan mengevaluasi proyek data mining. Metode ini telah digunakan secara luas dalam industri dan dianggap sebagai standar industri independen untuk data mining dan merupakan salah satu standar untuk menghasilkan data driven decision making yang berkualitas. Metode CRISP-DM terdiri dari 6 (enam) tahap.

IV.1 Business Understanding Phase

Penelitian ini dimulai dengan menentukan tujuan penelitian yaitu untuk mendeteksi tingkat fokus pengguna berdasarkan citra wajah mereka. Tujuan utama dari penelitian ini adalah untuk membangun sistem yang dapat secara real-time mendeteksi apakah seseorang fokus atau tidak berdasarkan ekspresi wajahnya. Penelitian ini sangat relevan untuk berbagai aplikasi, termasuk pendidikan jarak jauh, pemantauan kerja, dan interaksi pengguna dengan perangkat IoT.

IV.2 Data Understanding Phase

Fase ini melibatkan pengumpulan dan pemahaman data yang relevan untuk mendeteksi fokus. Data yang digunakan dalam penelitian ini adalah citra wajah yang menunjukkan kondisi fokus dan tidak fokus. Batasan dalam deteksi kondisi "*focused*" dan "*not focused*" adalah bahwa model akan menganggap seseorang tidak fokus ketika kepala mereka menghadap ke atas atau ke bawah (sudut vertikal), atau menengok ke kanan atau kiri (sudut horizontal). Sudut pengambilan gambar wajah sangat mempengaruhi keakuratan hasil ini. Jika gambar diambil dari sudut yang kurang tepat, seperti dari bawah atau terlalu miring, model mungkin kesulitan mendeteksi posisi kepala dengan benar, sehingga berpotensi menghasilkan deteksi yang tidak akurat. Oleh karena itu, pengambilan gambar harus dilakukan dari sudut yang sesuai untuk memastikan akurasi dalam deteksi fokus.



Gambar IV.1 Visualisasi Dataset

Pada Gambar IV.1 Plot ini mengilustrasikan distribusi dua kelompok data berdasarkan dua kondisi: "*focused*" dan "*not focused*." Titik-titik berwarna kuning-hijau mewakili data yang dikategorikan sebagai "*focused*," yang menunjukkan bahwa subjek berada dalam keadaan fokus. Titik-titik ini tersebar di sebagian besar area plot, menunjukkan variasi dalam distribusi ketika subjek sedang fokus. Titik-titik hijau mewakili data yang dikategorikan sebagai "*not focused*," yang menunjukkan bahwa subjek berada dalam keadaan tidak fokus. Titik-titik ini juga tersebar di seluruh plot, menunjukkan bahwa subjek masih dapat melakukan atau berperilaku dengan cara tertentu bahkan ketika tidak fokus.

Gambar IV.2 Sampel Dataset

Pada Gambar IV.2 dapat terlihat bahwa pengambilan sampel kecil dari satu kelas saja dapat memberikan banyak informasi tentang dataset. Variasi dalam latar belakang, kecerahan, orientasi, dan banyak lagi dapat menyebabkan beberapa

masalah dalam deep learning yang dapat kita antisipasi hanya dengan melihat sampel dari dataset ini.

IV.3 Data Preparation

Dalam penelitian, tahap persiapan data (data preparation) umumnya dibagi menjadi tiga bagian utama, yaitu splitting data, image preprocessing dan augmentation data.

IV.3.1 Splitting Data

Data yang telah terkumpul dibagi menjadi dua bagian, yaitu data untuk pelatihan (*training*) dan pengujian (*testing*). Data pelatihan memiliki proporsi yang lebih besar daripada data pengujian. Penelitian ini menggunakan 211 data pelatihan dan 63 data pengujian, yang terbagi ke dalam 2 kelas, yaitu *focused* dan *not focused*. Contoh data gambar yang digunakan pada penelitian ini dilampirkan pada Tabel IV1.

Tabel IV.1 Dataset

Label	Gambar
Focused	
Not Focused	

IV.3.2 Image Pre-processing

Image Preprocessing di *Edge Impulse* melibatkan teknik konversi ke *grayscale*, yang digunakan untuk mengurangi kompleksitas warna dalam citra dan memfokuskan pada informasi intensitas. Konversi ke *grayscale* dapat membantu

meningkatkan efisiensi model dan mengurangi jumlah data yang perlu diproses. Tahapan *pre-processing* ini menggunakan *library* sebagai berikut:

a) **numpy**: Digunakan untuk melakukan operasi numerik efisien pada *array* dan matriks. Dalam konteks pemrosesan gambar, numpy sering digunakan untuk mengubah gambar menjadi *array* numerik dan untuk manipulasi data.
b) **matplotlib.pyplot**: Modul *plotting* dari *library* matplotlib yang menyediakan antarmuka seperti MATLAB untuk membuat plot dan visualisasi data lainnya. **PIL (Python Imaging Library)**: Pustaka yang menyediakan berbagai alat untuk mengolah gambar. Dua fungsi utama yang digunakan adalah: *Image.open*: digunakan untuk memuat gambar dari direktori. *ImageOps.grayscale*: Mengonversi gambar berwarna menjadi *grayscale*.
c) **edge_impulse_sdk**: Library yang menyediakan berbagai alat dan fungsionalitas untuk mengintegrasikan model *machine learning* dengan perangkat keras *Edge Impulse*.

Langkah-langkah *pre-processing* dimulai dengan mengubah ukuran gambar menjadi 48x48 piksel. Kemudian, gambar tersebut dikonversi menjadi *grayscale* untuk mengurangi kompleksitas warna dan memfokuskan pada intensitas cahaya. Gambar yang telah diproses kemudian diubah menjadi *array numpy* untuk digunakan sebagai input ke model *machine learning*. kode *pre-processing* gambar di *Edge Impulse* :

```
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image, ImageOps

# Memuat gambar dari direktori
img = Image.open('path_to_image.jpg')

# Mengubah ukuran gambar menjadi 48x48 piksel
img_resized = img.resize((48, 48))

# Mengonversi gambar menjadi grayscale
img_gray = ImageOps.grayscale(img_resized)

# Mengonversi gambar menjadi array numpy
img_array = np.array(img_gray)

# Menampilkan gambar yang telah diproses
plt.imshow(img_array, cmap='gray')
plt.show()
```

IV.3.3 Augmentasi Data

Dataset citra untuk pelatihan diperbesar dengan teknik augmentasi data. Augmentasi data ini melibatkan transformasi seperti *flipping* (membalik gambar secara horizontal), *resizing* (mengubah ukuran gambar), *cropping* (memotong gambar), dan variasi pencahayaan pada citra. Hal ini membantu model mengenali objek dan pola dalam berbagai kondisi, sehingga meningkatkan akurasi dan mencegah *overfitting.* Pada tahap augmentasi data, digunakan beberapa fungsi dari *library* TensorFlow. Gambar secara acak akan diputar secara horizontal menggunakan tf.image.random_flip_left_right. Hal ini membantu model untuk mengenali objek meskipun posisinya dibalik. Gambar akan diperbesar ukurannya dan kemudian dipotong secara acak kembali ke dimensi aslinya. Hal ini dilakukan untuk mensimulasikan berbagai skenario di mana objek mungkin muncul lebih besar atau lebih kecil dalam gambar. Kecerahan gambar akan bervariasi secara acak menggunakan tf.image.random_brightness. Teknik ini membantu model beradaptasi dengan variasi pencahayaan yang mungkin terjadi dalam gambar nyata. Berikut adalah keseluruhan implementasi fungsi augmentasi :

```
def augment_image(image, label):
    # Flips the image randomly
    image = tf.image.random_flip_left_right(image)

    # Increase the image size, then randomly crop it
down to
    # the original dimensions
    resize_factor = random.uniform(1, 1.2)
    new_height = math.floor(resize_factor *
INPUT_SHAPE[0])
    new_width = math.floor(resize_factor *
INPUT_SHAPE[1])
    image = tf.image.resize_with_crop_or_pad(image,
new_height, new_width)
    image = tf.image.random_crop(image,
size=INPUT_SHAPE)

    # Vary the brightness of the image
    image = tf.image.random_brightness(image,
max_delta=0.2)

    return image, label

train_dataset = train_dataset.map(augment_image,
num_parallel_calls=tf.data.AUTOTUNE)
```

Langkah-langkah augmentasi ini dilakukan pada setiap *batch* data selama pelatihan model. Pengaturan ukuran batch menjadi 256, yang merupakan jumlah sampel yang diproses sebelum model diperbarui. Berikut adalah konfigurasi batch:

```
BATCH_SIZE = 256
train_dataset = train_dataset.batch(BATCH_SIZE,
drop_remainder=False).prefetch(tf.data.AUTOTUNE)
validation_dataset =
validation_dataset.batch(BATCH_SIZE,
drop_remainder=False).prefetch(tf.data.AUTOTUNE)
```

IV.4 Modeling

Tahap pemodelan melibatkan penerapan algoritma MobileNetV2 yang telah dilatih sebelumnya, sebuah varian dari *Convolutional Neural Network* (CNN). Model dibangun secara bertahap dengan lapisan-lapisan sebagai berikut:

1. Model Dasar :
   a) MobileNetV2 dengan input shape (48, 48, 1) (gambar *greyscale* berukuran 48x48 piksel).
   b) Lapisan model dasar dibekukan (tidak dapat dilatih) pada tahap awal.
2. Lapisan Tambahan :
   a) *Input Layer*: Lapisan input untuk menerima data citra dengan bentuk (48, 48, 1).
   b) *Model Base*: Mengambil output dari MobileNetV2 tanpa lapisan atas (*top layers).*
   c) *Reshape*: Mereshape output dari *model base* menjadi bentuk yang sesuai untuk lapisan berikutnya.
   d) *Dense Layer*: Lapisan *fully connected* dengan 8 unit dan aktivasi ReLU.
   e) *Dropout*: Dropout dengan rate 0.1 untuk regularisasi dan mencegah *overfitting.*
   f) *Flatten*: Meratakan output dari lapisan sebelumnya untuk input ke lapisan *fully connected.*
   g) Output Layer: Lapisan softmax untuk menghasilkan probabilitas kelas.
3. Kompilasi Model
   a) Model dikompilasi menggunakan optimizer Adam dengan *learning rate* 0.001 dan *loss function categorical_crossentropy.*

b) Model dilatih dengan dataset yang telah diaugmentasi selama 80 epoch.

4. Fine-Tuning :

a) Setelah pelatihan awal, model terbaik dari pelatihan tersebut dimuat kembali dan dilakukan *fine-tuning.*

b) Persentase tertentu dari lapisan model dasar diizinkan untuk dilatih kembali sementara lapisan lainnya tetap dibekukan.

c) Fine-tuning dilakukan selama 10 *epoch* tambahan dengan learning rate 0.000045 untuk mengoptimalkan kinerja model.

```
import math, requests
from pathlib import Path
import tensorflow as tf
from tensorflow.keras import Model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (
    Dense, InputLayer, Dropout, Reshape)
from tensorflow.keras.optimizers.legacy import Adam

sys.path.append('./resources/libraries')
import ei_tensorflow.training

WEIGHTS_PATH = './transfer-learning-
weights/edgeimpulse/MobileNetV2.0_35.96x96.grayscale.bsiz
e_64.lr_0_005.epoch_260.val_loss_3.10.val_accuracy_0.35.h
df5'

# Download the model weights
root_url = 'https://cdn.edgeimpulse.com/'
p = Path(WEIGHTS_PATH)
if not p.exists():
    print(f"Pretrained weights {WEIGHTS_PATH}
unavailable; downloading...")
    if not p.parent.exists():
        p.parent.mkdir(parents=True)
    weights_data = requests.get(root_url +
WEIGHTS_PATH[2:]).content
    with open(WEIGHTS_PATH, 'wb') as f:
        f.write(weights_data)
    print(f"Pretrained weights {WEIGHTS_PATH}
unavailable; downloading OK")
    print("")
```

```
INPUT_SHAPE = (48, 48, 1)

base_model = tf.keras.applications.MobileNetV2(
    input_shape=INPUT_SHAPE, alpha=0.35,
    weights=WEIGHTS_PATH
)

base_model.trainable = False

model = Sequential()
model.add(InputLayer(input_shape=INPUT_SHAPE,

name='x_input'))
# Don't include the base model's top layers
last_layer_index = -3
model.add(Model(inputs=base_model.inputs,
outputs=base_model.layers[last_layer_index].output))
model.add(Reshape((-1, model.layers[-
1].output.shape[3])))
model.add(Dense(8, activation='relu'))
model.add(Dropout(0.1))
model.add(Flatten())
model.add(Dense(classes, activation='softmax'))

model.compile(optimizer=tf.keras.optimizers.Adam(learning
_rate=LEARNING_RATE),
                loss='categorical_crossentropy',
                metrics=['accuracy'])

model.fit(train_dataset,
validation_data=validation_dataset, epochs=EPOCHS,
verbose=2, callbacks=callbacks)

print('')
print('Initial training done.', flush=True)

# How many epochs we will fine tune the model
FINE_TUNE_EPOCHS = 10
# What percentage of the base model's layers we will fine
tune
FINE_TUNE_PERCENTAGE = 65

print('Fine-tuning best model for {}
epochs...'.format(FINE_TUNE_EPOCHS), flush=True)

# Load best model from initial training
model =
ei_tensorflow.training.load_best_model(BEST_MODEL_PATH)
```

```
# Determine which layer to begin fine tuning at
model_layer_count = len(model.layers)
fine_tune_from = math.ceil(model_layer_count * ((100 -
FINE_TUNE_PERCENTAGE) / 100))

# Allow the entire base model to be trained
model.trainable = True
# Freeze all the layers before the 'fine_tune_from' layer
for layer in model.layers[:fine_tune_from]:
    layer.trainable = False

model.compile(optimizer=tf.keras.optimizers.Adam(learning
_rate=0.000045),
                loss='categorical_crossentropy',
                metrics=['accuracy'])

model.fit(train_dataset,
                epochs=FINE_TUNE_EPOCHS,
                verbose=2,
                validation_data=validation_dataset,
                callbacks=callbacks,
                class_weight=None
            )
```

BAB V Perancangan dan Implementasi

V.1 Skenario Pengujian Model

Pengujian dalam penelitian ini dilakukan melalui empat tahap yang berbeda, masing-masing dirancang untuk mengevaluasi kinerja model dengan cara yang lebih mendalam dan menyeluruh. Setiap tahap pengujian mencakup penggunaan sejumlah *epoch* yang berbeda, yang bertujuan untuk menguji sejauh mana model dapat mengoptimalkan proses pelatihan dengan variasi jumlah iterasi yang digunakan. Penggunaan *epoch* yang berbeda-beda ini penting untuk mengevaluasi bagaimana model bereaksi terhadap variasi waktu pelatihan, serta untuk menilai kemampuan model dalam menggeneralisasi data yang belum pernah dilihat sebelumnya. Rincian lebih lanjut mengenai setiap tahapan pengujian, termasuk variasi *epoch* yang diterapkan dan parameter lainnya yang digunakan, akan dijelaskan lebih lanjut pada uraian berikut ini.

A. Pengujian ke-1

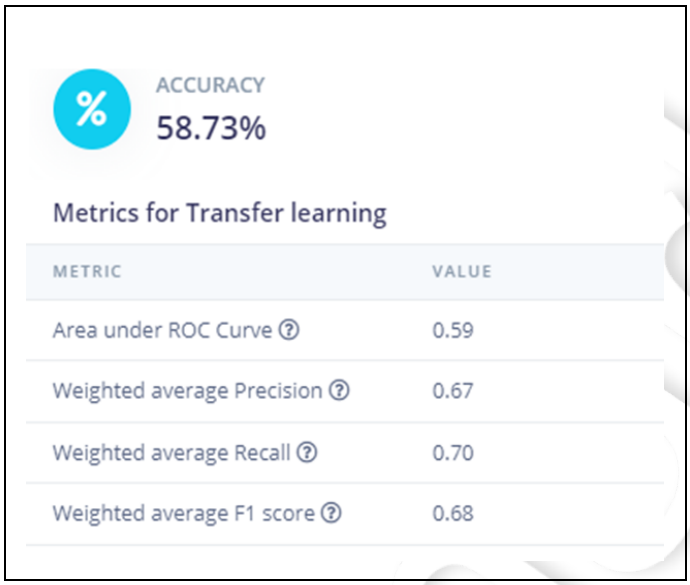
Untuk pengujian 1 dengan ukuran *batch* 64 dan *epoch* dengan jumlah 40. Pengujian dilakukan dengan menggunakan data uji sebanyak 63 citra. Pengujian ini menghasilkan tabel *confusion matrix* yang disajikan Tabel 5.1.

Tabel V.1 Hasil pengujian ke-1 *confusion matrix epoch* 40

	FOCUSED	NOT FOCUSED	UNCERTAIN
FOCUSED	21.1%	57.9%	21.1%
NOT FOCUSED	9.1%	75%	15.9%
F1 SCORE	0.30	0.75	

Berdasarkan Tabel V.1, dapat dilihat bahwa model ini berhasil mengklasifikasikan 37 dari 63 data uji citra dengan benar, sementara 26 data lainnya diklasifikasikan ke kelas yang salah.. Dari hasil tersebut didapatkan 58,73% untuk nilai akurasi, 59% untuk nilai AUC, 67% untuk nilai presisi, 70% untuk nilai recall, dan 68%

untuk nilai F1 *score*. Nilai - nilai tersebut ditunjukkan pada gambar berikut. N Nilai-nilai tersebut dapat dilihat pada Gambar V.1.



Gambar V.1 Evaluasi pengujian ke-1 epoch 40

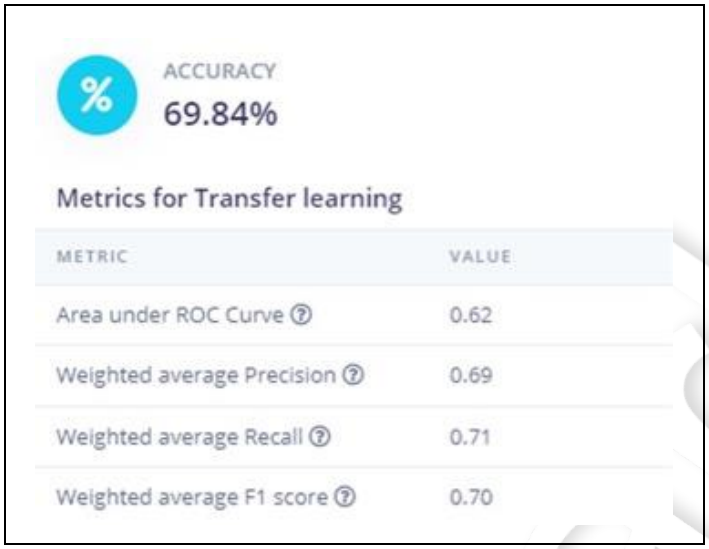
B. Pengujian ke-2

Untuk pengujian ke 2 dengan ukuran *batch* 64 dan *epoch* dengan jumlah 45. Pengujian dilakukan dengan menggunakan data uji sebanyak 63 citra. Pengujian ini menghasilkan tabel *confusion matrix* yang disajikan sebagai berikut.

Tabel V.2 Hasil pengujian ke-2 *confusion matrix epoch* 45

	FOCUSED	NOT FOCUSED	UNCERTAIN
FOCUSED	36.8%	52.6%	10.5%
NOT FOCUSED	11.4%	84.1%	4.5%
F1 SCORE	0.45	0.81	

Berdasarkan Tabel V.2 dapat dlihat model ini berhasil mengklasifikan 44 dari 63 data uji citra dengan benar. Sisanya sebanyak 19 data yang diklasifikasikan pada kelas yang salah. Dari hasil tersebut didapatkan 69,84% untuk nilai akurasi, 62% untuk nilai AUC, 69% untuk nilai presisi, 71% untuk nilai recall, dan 70% untuk nilai F1 *score*. Nilai - nilai tersebut ditunjukkan pada Gambar V.2.



Gambar V.2 Evaluasi pengujian ke-2 epoch 45

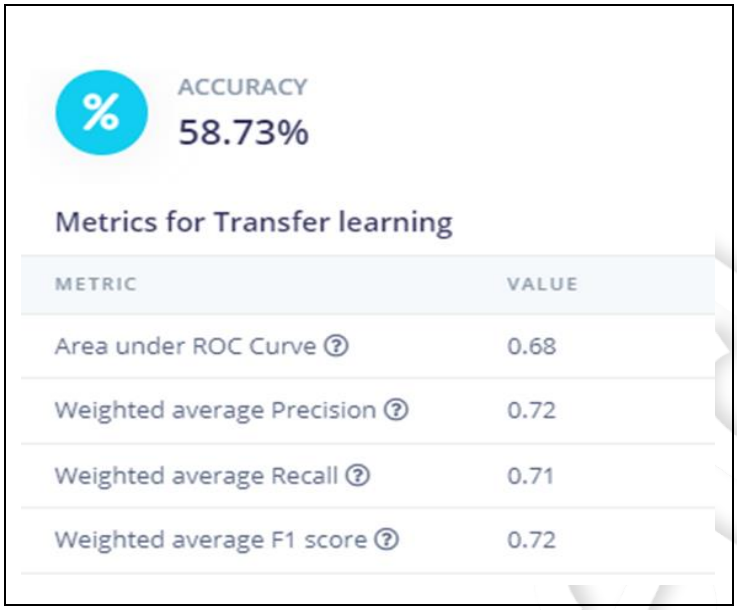
C. Pengujian ke-3

Untuk pengujian 3 dengan ukuran *batch* 64 dan *epoch* dengan jumlah 50. Pengujian dilakukan dengan menggunakan data uji sebanyak 63. Pengujian ini menghasilkan tabel *confusion matrix* yang disajikan pada Tabel V.3.

Tabel V.3 Hasil pengujian ke-3 *confusion matrix epoch* 50

	FOCUSED	NOT FOCUSED	UNCERTAIN
FOCUSED	47.4%	36.8%	15.8%
NOT FOCUSED	13.6%	63.6%	22.7%
F1 SCORE	0.53	0.71	

Berdasarkan Tabel V.3 dapat dilihat model ini berhasil mengklasifikan 37 dari 63 data uji citra dengan benar. Sisanya sebanyak 26 data yang diklasifikasikan pada kelas yang salah. Dari hasil tersebut didapatkan 58,73% untuk nilai akurasi, 68% untuk nilai AUC, 72% untuk nilai presisi, 71% untuk nilai recall, dan 72% untuk nilai F1 *score*. Nilai - nilai tersebut ditunjukkan pada Gambar V.3.



Gambar V.3 Evaluasi pengujian ke-3 epoch 50

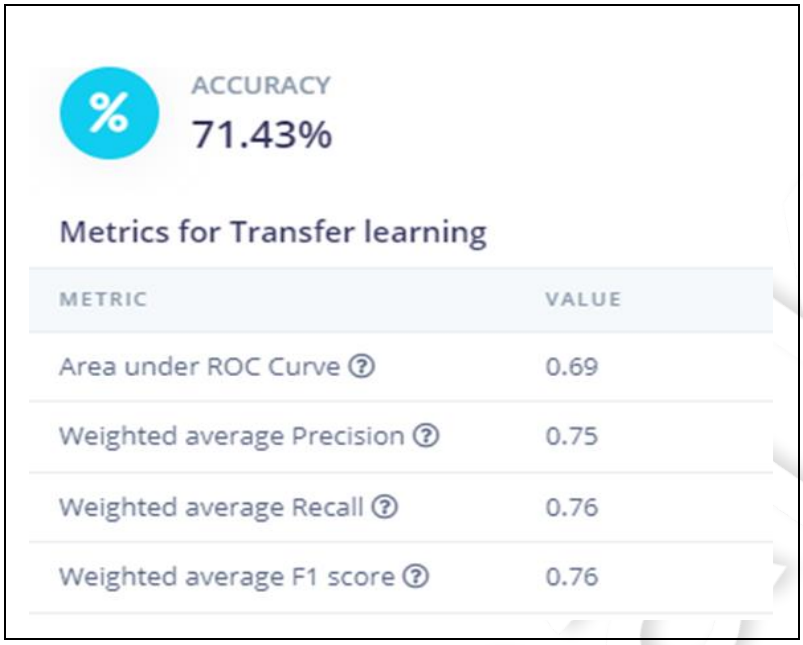
D. Pengujian ke-4

Untuk pengujian 4 dengan ukuran *batch* 64 dan *epoch* dengan jumlah 55. Pengujian dilakukan dengan menggunakan data uji sebanyak 63. Pengujian ini menghasilkan tabel *confusion matrix* yang disajikan Tabel V.4.

Tabel V.4 Hasil pengujian ke-4 *confusion matrix epoch* 55

	FOCUSED	NOT FOCUSED	UNCERTAIN
FOCUSED	52.6%	42.1%	5.3%
NOT FOCUSED	13.6%	79.5%	6.8%
F1 SCORE	0.57	0.80	

Berdasarkan Tabel V.4 dapat dlihat model ini berhasil mengklasifikan 45 dari 63 data uji citra dengan benar. Sisanya sebanyak 18 data yang diklasifikasikan pada kelas yang salah. Dari hasil tersebut didapatkan 71,43% untuk nilai akurasi, 69% untuk nilai AUC, 75% untuk nilai presisi, 76% untuk nilai recall, dan 76% untuk nilai F1 *score.* Nilai - nilai tersebut ditunjukkan pada Gambar V.4.



Gambar V.4 Evaluasi pengujian ke-4 epoch 55

Dengan berbagai ukuran *epoch* (40, 45, 50, dan 55) dan ukuran *batch* yang tetap, beberapa hal dapat dianalisis terkait kinerja model. Pada *epoch* 40, model menunjukkan akurasi 58,73%, AUC 59%, presisi 67%, recall 70%, dan F1 score 68%. Ini menunjukkan bahwa model mampu mengklasifikasikan 37 dari 63 citra uji dengan benar, tetapi masih ada 26 citra yang salah klasifikasi. Hasil ini menunjukkan bahwa model belum optimal, dengan akurasi yang masih rendah dan AUC yang tidak terlalu baik, menandakan bahwa kemampuan model dalam membedakan kelas masih lemah.

Saat epoch ditingkatkan menjadi 45, terjadi peningkatan kinerja dengan akurasi 69,84%, AUC 62%, presisi 69%, recall 71%, dan F1 score 70%. Pada tahap ini, model berhasil mengklasifikasikan 44 dari 63 citra uji dengan benar, dengan hanya 19 citra yang salah klasifikasi. Ini menunjukkan adanya perbaikan yang signifikan dalam kemampuan model, khususnya dalam membedakan kelas yang berbeda, meskipun nilai AUC masih menunjukkan ruang untuk peningkatan lebih lanjut.

Namun, pada epoch 50, meskipun nilai AUC meningkat menjadi 68%, akurasi menurun kembali menjadi 58,73%, dengan presisi 72%, recall 71%, dan F1 score 72%. Model kembali hanya mampu mengklasifikasikan 37 dari 63 citra uji dengan benar, serupa dengan hasil pada epoch 40. Ini mengindikasikan bahwa model mungkin mengalami *overfitting* atau variasi dalam kinerja karena perubahan *epoch*.

Akhirnya, pada *epoch* 55, kinerja model meningkat dengan akurasi 71,43%, AUC 69%, presisi 75%, recall 76%, dan F1 score 76%. Model berhasil mengklasifikasikan 45 dari 63 citra uji dengan benar, menunjukkan bahwa peningkatan jumlah epoch dapat memperbaiki kinerja model secara keseluruhan, terutama dalam hal akurasi dan kemampuan deteksi kelas yang benar.

Secara keseluruhan, meskipun peningkatan *epoch* dapat memperbaiki beberapa metrik kinerja, hasilnya tidak selalu konsisten, dan model mungkin mengalami fluktuasi dalam kinerja yang perlu diwaspadai.

V.2 Deployment

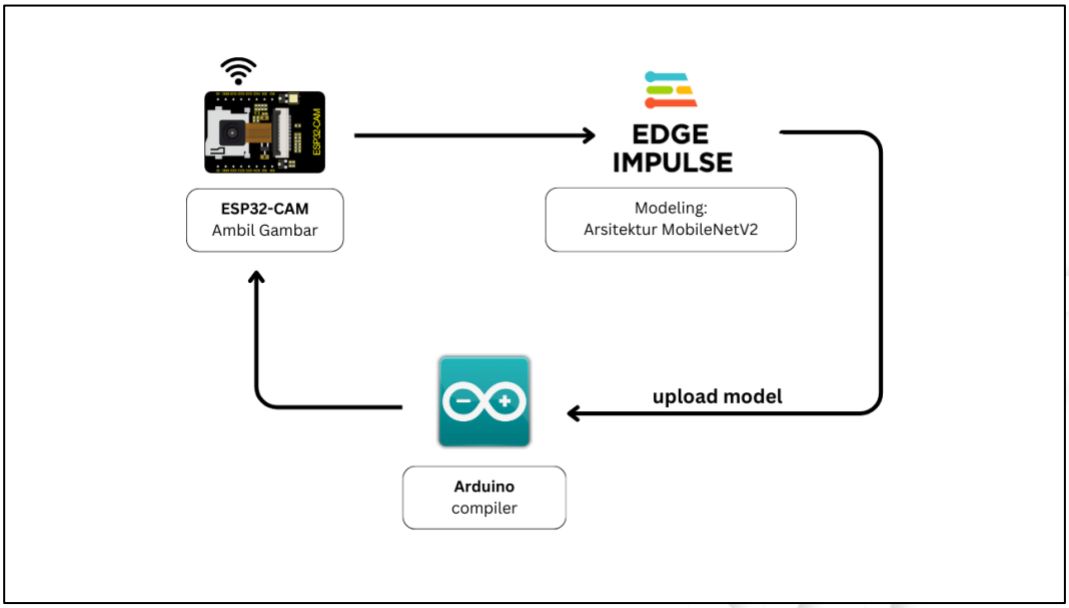
V.2.1 Arsitektur Sistem

Arsitektur sistem yang diterapkan dalam penelitian ini dibagi menjadi dua konsep utama, yaitu Arsitektur Pengembangan (*Development Architecture*) dan Arsitektur Operasional (*Operational Architecture).*

A. Arsitektur Pengembangan

Arsitektur pengembangan berkaitan dengan struktur dan desain sistem pada tahap pengembangan, yang mencakup perencanaan, perancangan, dan pembuatan perangkat lunak atau sistem sebelum diterapkan dalam lingkungan operasional. Fokus utama dari arsitektur ini adalah bagaimana sistem dibangun dan dikembangkan, termasuk pemilihan teknologi, framework, serta bahasa pemrograman yang sesuai dengan kebutuhan aplikasi. Selain itu, arsitektur pengembangan juga mencakup proses pengujian untuk memastikan bahwa sistem yang dibangun dapat berfungsi sesuai dengan spesifikasi dan persyaratan yang telah ditetapkan. Arsitektur pengembangan berfokus pada aspek fungsionalitas, modularitas, dan skalabilitas, yang bertujuan untuk meminimalisir risiko kesalahan dan meningkatkan kualitas produk sebelum sistem diterapkan pada tahap operasional.

Pada tahap pengembangan penelitian ini, dilakukan pengembangan model yang dapat mendeteksi citra wajah dan mengkategorikannya menjadi dua kelas, yaitu fokus dan tidak fokus. Hal ini dapat dilihat pada Gambar 5.5.



Gambar V.5 Arsitektur Pengembangan

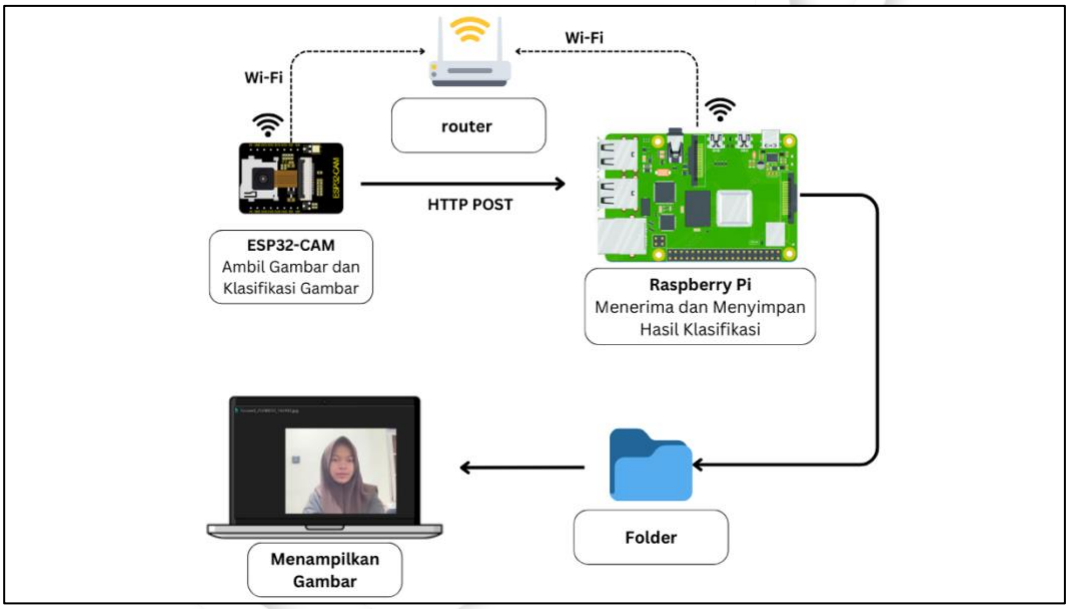
Pada Gambar 5.5 ditunjukkan arsitektur sistem yang terdiri dari beberapa komponen utama, yaitu pengambilan gambar, pembuatan model, dan pengiriman model. ESP32-CAM berfungsi sebagai perangkat untuk pengambilan gambar serta menerima model yang telah dilatih. Model deteksi fokus/tidak fokus dikembangkan menggunakan kinerja Edge Impulse dengan menerapkan arsitektur *Convolutional Neural Network* (CNN) berbasis MobileNetV2. Hasil model yang telah dilatih kemudian diunggah ke ESP32-CAM melalui Arduino IDE dengan pemrograman menggunakan bahasa C, yang memungkinkan sistem beroperasi dalam konteks aplikasi nyata.

B. Arsitektur Operasional

Arsitektur operasional berkaitan dengan struktur dan desain sistem yang telah diterapkan dan dijalankan dalam lingkungan produksi. Fokus utama dari arsitektur ini adalah pada bagaimana sistem atau aplikasi berfungsi secara efektif dan efisien setelah diterapkan, serta bagaimana ia beroperasi dalam kondisi nyata. Arsitektur operasional tidak hanya mencakup pemeliharaan sistem, tetapi juga pengelolaan dan pemantauan kinerja sistem secara berkelanjutan selama fase operasional. Hal ini meliputi pengelolaan sumber daya, seperti server dan infrastruktur jaringan, untuk memastikan ketersediaan dan kinerja yang optimal. Selain itu, arsitektur

operasional juga mencakup pengelolaan masalah yang mungkin muncul, termasuk penanganan kesalahan atau gangguan, serta pembaruan perangkat lunak atau perangkat keras yang diperlukan untuk memastikan kelangsungan operasional. Dalam konteks ini, arsitektur operasional memastikan bahwa sistem dapat diandalkan, aman, dan dapat berkembang seiring waktu, sambil tetap memenuhi kebutuhan pengguna dan memastikan keberlangsungan operasional dalam jangka panjang.

Pada tahap operasional penelitian ini, dilakukan serangkaian proses yang mencakup pengambilan gambar, klasifikasi gambar, pengiriman gambar ke server, serta penampilan gambar secara otomatis pada folder di layar komputer. Proses tersebut dapat dilihat pada Gambar 5.6.



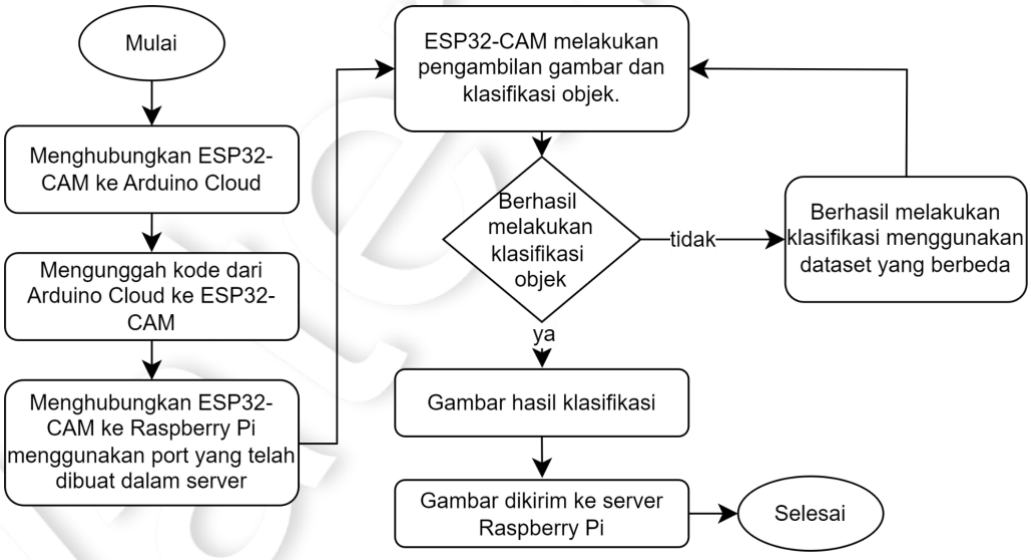
Gambar V.6 Arsitektur Operasional

Sistem ini terdiri dari beberapa komponen utama yang bekerja secara sinergis untuk menangkap, mengirimkan, dan menampilkan gambar secara otomatis. ESP32-CAM, sebuah modul kamera dengan integrasi Wi-Fi, berfungsi untuk menangkap gambar secara *real-time* dan mengirimkannya melalui jaringan Wi-Fi. Router berperan sebagai pusat jaringan yang menghubungkan semua perangkat dalam sistem, serta meneruskan data yang dikirimkan oleh ESP32-CAM ke Raspberry Pi. Raspberry Pi, yang merupakan komputer mini, bertindak sebagai pengontrol utama dalam sistem, menjalankan program Python untuk menerima data gambar dari

ESP32-CAM melalui protokol HTTP POST, dan menyimpannya di penyimpanan lokal. *Python* digunakan sebagai bahasa pemrograman untuk mengontrol Raspberry Pi, memproses data gambar, serta memungkinkan pengembangan program yang mengotomatiskan tugas-tugas seperti menerima dan menyimpan data gambar. Laptop kemudian digunakan untuk menampilkan gambar yang telah ditangkap dan disimpan oleh sistem, baik secara *real-time* maupun berdasarkan permintaan. Dengan demikian, sistem ini memungkinkan pengumpulan data gambar secara real-time, pemantauan jarak jauh melalui perangkat yang terhubung ke jaringan, serta otomatisasi proses pengambilan, pengiriman, dan penyimpanan gambar tanpa memerlukan intervensi manusia secara berkelanjutan.

V.2.2 Demonterasi *Live Classification* pada ESP32-CAM

Penelitian ini menggabungkan perhitungan Pembelajaran Mesin pada ESP32-CAM, yang menangkap dan mengklasifikasikan gambar menggunakan model yang diterapkan. Hasil klasifikasi kemudian disimpan dan dipantau secara real-time di server Raspberry Pi. Integrasi ini memungkinkan pemrosesan dan visualisasi data yang efisien melalui Raspberry Pi. Untuk diagram alur sistem, pada Gambar V.7.



Gambar V.7 Demonterasi *Live Classification* pada ESP32-CAM

Pada Gambar V.7 merupakan *flowchart* proses cara kerja klasifikasi objek menggunakan ESP32-CAM. Proses ini dimulai dengan menghubungkan modul

ESP32-CAM ke *Arduino Cloud*, kode di unggah ke perangkat. Setelah terkoneksi, kode yang dirancang untuk pengambilan gambar dan klasifikasi objek diunggah ke ESP32-CAM. Setelah pengunggahan kode selesai, ESP32-CAM mulai beroperasi dengan mengambil gambar dan menjalankan algoritma klasifikasi pada gambar yang diperoleh. Pada tahap evaluasi, sistem menentukan keberhasilan proses klasifikasi. Jika klasifikasi tidak berhasil, maka sistem akan mencoba kembali dengan menggunakan data yang berbeda. Namun, jika klasifikasi berhasil, maka gambar hasil klasifikasi akan disimpan atau ditampilkan sebagai output. Langkah selanjutnya melibatkan pengiriman gambar yang telah diklasifikasikan ke server Raspberry Pi melalui *port* yang telah ditentukan. Raspberry Pi berfungsi sebagai pusat penyimpanan atau pemrosesan lanjutan dari gambar tersebut. Setelah pengiriman gambar selesai, proses keseluruhan dianggap selesai.

Untuk memantau hasil klasifikasi secara real-time, Raspberry Pi digunakan sebagai server monitoring. Data klasifikasi dari ESP32-CAM dikirim ke Raspberry Pi melalui jaringan yang terhubung. Dengan sistem ini, hasil klasifikasi dapat dipantau dan dianalisis dari mana saja selama terhubung ke jaringan yang sama atau melalui akses remote ke Raspberry Pi.

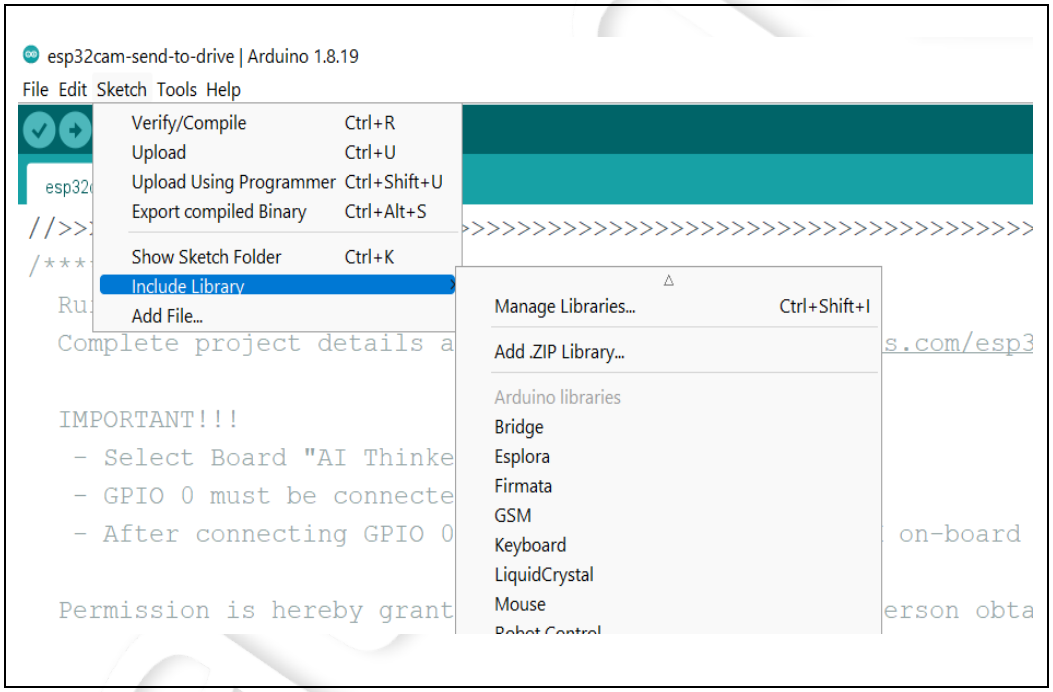
Pada penelitian ini, ESP32-CAM digunakan untuk mengambil gambar dan mengklasifikasikannya secara otomatis. Mengingat keterbatasan memori dan penyimpanan lokal yang dimiliki oleh ESP32-CAM, pengiriman data secara langsung ke server diperlukan. Hal ini dikarenakan penyimpanan hasil klasifikasi gambar secara lokal pada perangkat tersebut tidak memungkinkan untuk dilakukan dalam jangka panjang. Dengan mengirimkan data ke server, gambar yang diambil dapat disimpan dalam sistem penyimpanan yang lebih besar dan dikelola secara lebih efisien. Selain itu, pengiriman data ke server memungkinkan pengelolaan dan pemantauan yang lebih baik, terutama dalam konteks aplikasi yang lebih besar, seperti analisis data secara historis maupun secara real-time. Selain itu, ESP32-CAM beroperasi dalam jaringan intranet yang terbatas, sehingga tidak dapat diakses secara langsung oleh publik. Oleh karena itu, perangkat ini mengirimkan data atau informasi yang diperolehnya ke server, sehingga dapat diakses oleh pengguna yang berada di luar jaringan lokal.

V.2.3 Implementasi Model dari Edge Impulse ke ESP32CAM

Implementasi model dari *Edge Impulse* ke ESP32-CAM melibatkan beberapa langkah penting untuk memastikan model yang dilatih di *Edge Impulse* dapat dijalankan dan digunakan secara efektif di perangkat ESP32-CAM. Setelah mengekspor model dari *Edge Impulse* dan mempersiapkan ESP32-CAM, langkah selanjutnya adalah mengimplementasikan model tersebut pada ESP32-CAM.

V.2.4 Memasukkan model

Langkah pertama adalah mengekspor model dari Edge Impulse dan mempersiapkan ESP32-CAM, Cara menyisipkan model yang diekspor adalah dengan menambahkan library model dalam bentuk .zip pada Arduino IDE, seperti ditunjukkan pada Gambar V.8



Gambar V.8 Menginstalasi model

V.2.5 Implementasi Kode

Pada pengimplementasian kode, ada beberapa library yang digunakan untuk menjalankan program ini, yaitu:

```
#include <iot_edge-computing_inferencing.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include <base64.h>
#include "esp_camera.h"
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "driver/rtc_io.h"
#include "edge-impulse-sdk/dsp/image/image.hpp"
```

Kode di atas dirancang untuk mendukung penelitian deteksi fokus dan tidak fokus menggunakan machine learning pada ESP32-CAM dan server Raspberry Pi. Dimulai dengan mengimpor berbagai library penting, kode ini menggunakan iot_edge-computing_inferencing.h untuk inferensi model *machine learning* dari Edge Impulse, WiFi.h untuk koneksi ke jaringan Wi-Fi, dan HTTPClient.h untuk melakukan HTTP requests guna mengirim data ke server. Selain itu, ArduinoJson.h digunakan untuk memproses data dalam format JSON, base64.h untuk meng-encode gambar dalam format base64 sebelum dikirim, dan esp_camera.h untuk mengontrol kamera ESP32-CAM. Library tambahan seperti soc/soc.h, soc/rtc_cntl_reg.h, dan driver/rtc_io.h digunakan untuk pengaturan sistem dan pin kamera, sementara edge-impulse-sdk/dsp/image/image.hpp dari Edge Impulse menyediakan fungsi pemrosesan gambar.

Setelah *import library*, konfigurasi kamera diatur dengan mendefinisikan pin, format pixel, frekuensi clock, dan ukuran frame untuk memastikan kualitas gambar yang optimal.

```
// Konfigurasi kamera
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
```

```
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

#define EI_CAMERA_RAW_FRAME_BUFFER_COLS 320
#define EI_CAMERA_RAW_FRAME_BUFFER_ROWS 240
#define EI_CAMERA_FRAME_BYTE_SIZE 3

#define WLAN_SSID      "UINSGD-ClassRoom"
#define WLAN_PASS      "uinbandunghotspot"

// Konfigurasi server Raspberry Pi
#define RPI_SERVER      "http://10.200.45.9:5000"
#define RPI_ENDPOINT    "/upload"

WiFiClient client;
static bool debug_nn = false;
static bool is_initialised = false;
uint8_t *snapshot_buf;

static camera_config_t camera_config = {
  .pin_pwdn = PWDN_GPIO_NUM,
  .pin_reset = RESET_GPIO_NUM,
  .pin_xclk = XCLK_GPIO_NUM,
  .pin_sscb_sda = SIOD_GPIO_NUM,
  .pin_sscb_scl = SIOC_GPIO_NUM,

  .pin_d7 = Y9_GPIO_NUM,
  .pin_d6 = Y8_GPIO_NUM,
  .pin_d5 = Y7_GPIO_NUM,
  .pin_d4 = Y6_GPIO_NUM,
  .pin_d3 = Y5_GPIO_NUM,
  .pin_d2 = Y4_GPIO_NUM,
  .pin_d1 = Y3_GPIO_NUM,
  .pin_d0 = Y2_GPIO_NUM,
```

```
  .pin_vsync = VSYNC_GPIO_NUM,
  .pin_href = HREF_GPIO_NUM,
  .pin_pclk = PCLK_GPIO_NUM,

  .xclk_freq_hz = 20000000,
  .ledc_timer = LEDC_TIMER_0,
  .ledc_channel = LEDC_CHANNEL_0,

  .pixel_format = PIXFORMAT_JPEG,
  .frame_size = FRAMESIZE_QVGA,
  .jpeg_quality = 12,
  .fb_count = 1,
  .fb_location = CAMERA_FB_IN_PSRAM,
  .grab_mode = CAMERA_GRAB_WHEN_EMPTY,
};
```

Untuk memungkinkan ESP32-CAM terhubung dengan jaringan Wi-Fi dan berkomunikasi dengan Raspberry Pi, perlu dilakukan konfigurasi tertentu pada kode program. Konfigurasi ini meliputi pengaturan SSID (*Service Set Identifier*), password untuk mengakses jaringan Wi-Fi, serta alamat IP server Raspberry Pi dan endpoint untuk menerima dan memproses data yang dikirimkan oleh ESP32-CAM, seperti pada kode dibawah ini.

```
void setup() {
  Serial.begin(115200);
  Serial.println("Demo Inferensi Edge Impulse");

  if (!ei_camera_init()) {
    ei_printf("Gagal menginisialisasi Kamera!\r\n");
  } else {
    ei_printf("Kamera berhasil diinisialisasi\r\n");
  }

  pinMode(4, OUTPUT);
  digitalWrite(4, LOW);

  setup_wifi();
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
}
```

Fungsi setup() adalah menginisialisasi komunikasi serial dengan menggunakan fungsi Serial.begin(). Ini bertujuan untuk memulai komunikasi antara ESP32-CAM dan komputer atau perangkat lain yang terhubung melalui port serial. Pada umumnya, komunikasi serial digunakan untuk melakukan debugging atau untuk

menampilkan informasi status yang berguna selama pengembangan aplikasi. Dalam kasus ini, baud rate yang digunakan adalah 115200, yang merupakan standar kecepatan komunikasi serial yang banyak digunakan pada ESP32.

Setelah komunikasi serial siap, langkah selanjutnya adalah menginisialisasi kamera. Pada ESP32-CAM, fungsi ei_camera_init() bertanggung jawab untuk mengkonfigurasi dan mengaktifkan kamera. Setelah kamera berhasil diinisialisasi, langkah selanjutnya adalah menghubungkan ESP32-CAM ke jaringan Wi-Fi. Hal ini penting untuk mengaktifkan konektivitas internet atau komunikasi antara ESP32-CAM dengan perangkat lain, seperti server atau perangkat lain yang terhubung dalam jaringan yang sama. Fungsi yang digunakan untuk ini adalah setup_wifi(), yang bertugas untuk menghubungkan ESP32-CAM ke jaringan Wi-Fi yang telah ditentukan sebelumnya. Setelah koneksi berhasil, alamat IP yang diperoleh ditampilkan, dan pengaturan RTC dilakukan untuk menonaktifkan deteksi brownout. Proses ini memastikan bahwa ESP32-CAM dapat terhubung dengan jaringan, menangkap gambar, dan berinteraksi dengan perangkat lain dengan cara yang stabil dan efisien.

```
void loop() {
  snapshot_buf = (uint8_t
*)malloc(EI_CAMERA_RAW_FRAME_BUFFER_COLS *
EI_CAMERA_RAW_FRAME_BUFFER_ROWS *
EI_CAMERA_FRAME_BYTE_SIZE);

  if (snapshot_buf == nullptr) {
    ei_printf("ERR: Gagal mengalokasikan buffer
snapshot!\n");
    return;
  }

  ei::signal_t signal;
  signal.total_length = EI_CLASSIFIER_INPUT_WIDTH *
EI_CLASSIFIER_INPUT_HEIGHT;
  signal.get_data = &ei_camera_get_data;

  if (!ei_camera_capture((size_t)EI_CLASSIFIER_INPUT_WIDTH,
(size_t)EI_CLASSIFIER_INPUT_HEIGHT, snapshot_buf)) {
    ei_printf("Gagal menangkap gambar\r\n");
    free(snapshot_buf);
    return;
  }
```

```
  ei_impulse_result_t result = { 0 };
  EI_IMPULSE_ERROR err = run_classifier(&signal, &result,
debug_nn);

  if (err != EI_IMPULSE_OK) {
    ei_printf("ERR: Gagal menjalankan classifier (%d)\n",
err);
    free(snapshot_buf);
    return;
  }

  ei_impulse_result_t result = { 0 };
  EI_IMPULSE_ERROR err = run_classifier(&signal, &result,
debug_nn);

  if (err != EI_IMPULSE_OK) {
    ei_printf("ERR: Gagal menjalankan classifier (%d)\n",
err);
    free(snapshot_buf);
    return;
  }

  ei_printf("Prediksi (DSP: %d ms., Klasifikasi: %d ms.,
Anomali: %d ms.): \n",
            result.timing.dsp,
result.timing.classification, result.timing.anomaly);

  String labelStr = "";
  bool isFocused = true;

  for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++)
{
    ei_printf("    %s: %.5f\n",
result.classification[ix].label,
              result.classification[ix].value);
    if (result.classification[ix].value > 0.5) {
      labelStr = result.classification[ix].label;
      if (labelStr == "not focused") {
        isFocused = false;
      }
    }
  }

  if (isFocused) {
    labelStr = "focused";
  }
```

Di dalam fungsi loop(), buffer memori dialokasikan untuk menyimpan gambar yang diambil dari kamera. Fungsi ei_camera_capture() menangkap gambar, mengkonversinya ke format yang sesuai, dan jika diperlukan, meresize gambar untuk mencocokkan dimensi input model *machine learning*. Inferensi dilakukan dengan model Edge Impulse melalui fungsi run_classifier(), yang menghasilkan hasil klasifikasi apakah gambar tersebut fokus atau tidak fokus. Hasil klasifikasi dikumpulkan, dan gambar yang diambil di-encode dalam format base64 menggunakan base64.h. Fungsi send_image_and_label_to_rpi() mengirimkan data ke server Raspberry Pi, mengemas label hasil inferensi dan gambar dalam format JSON dan mengirimkan HTTP POST request ke server. Kode memeriksa kode respons untuk memastikan bahwa data dikirim dengan sukses.

Kode berikutnya adalah Fungsi setup_wifi() untuk memastikan ESP32-CAM terhubung ke jaringan Wi-Fi dan menampilkan alamat IP yang diperoleh setelah koneksi berhasil.

```
void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Menghubungkan ke ");
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi terhubung");
  Serial.println("Alamat IP: ");
  Serial.println(WiFi.localIP());
}
```

Fungsi ei_camera_init() menginisialisasi kamera dengan konfigurasi yang telah ditetapkan, mengatur parameter sensor untuk kualitas gambar yang optimal.

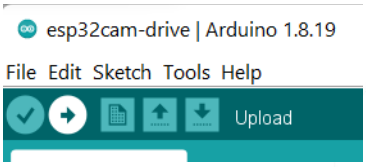
```
bool ei_camera_init(void) {
  if (is_initialised) return true;

  esp_err_t err = esp_camera_init(&camera_config);

  if (err != ESP_OK) {
    Serial.printf("Inisialisasi kamera gagal dengan error
0x%x\n", err);
    return false;
  }

  sensor_t *s = esp_camera_sensor_get();

  if (s->id.PID == OV3660_PID) {
    s->set_vflip(s, 1);       // membalikkan gambar
    s->set_brightness(s, 1);  // meningkatkan kecerahan
sedikit
    s->set_saturation(s, 0);  // menurunkan saturasi
  }

#if defined(CAMERA_MODEL_M5STACK_WIDE)
  s->set_vflip(s, 1);
  s->set_hmirror(s, 1);
#endif

  is_initialised = true;
  return true;
}
```

Fungsi ei_camera_capture() menangkap gambar, mengkonversinya, dan jika perlu, meresize gambar agar sesuai dengan dimensi input model machine learning. Dengan mengintegrasikan berbagai fungsi ini, kode memungkinkan pengambilan gambar, pemrosesan data, inferensi model, dan komunikasi jaringan secara efisien, mendukung deteksi fokus dan tidak fokus secara real-time dan pengiriman hasil deteksi ke server pusat untuk analisis lebih lanjut.

```
bool ei_camera_capture(uint32_t img_width, uint32_t
img_height, uint8_t *out_buf) {
  bool do_resize = false;

  if (!is_initialised) {
    ei_printf("ERR: Kamera belum diinisialisasi\r\n");
    return false;
  }

  digitalWrite(4, HIGH);
  camera_fb_t *fb = esp_camera_fb_get();
```

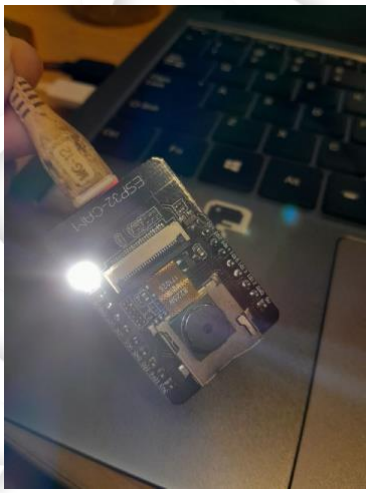
```
if (!fb) {
  ei_printf("Gagal menangkap gambar\n");
  return false;
}
```

Kemudian, jika kode sudah diimplementasikan pada Arduino IDE, maka langkah selanjutnya adalah mengupload kode pada ESP32CAM dengan mengklik tombol upload (panah kanan).



Gambar V.9 Tombol upload

Setelah ESP32-CAM terhubung ke jaringan dan kamera diinisialisasi, perangkat akan mulai menangkap gambar. Berdasarkan hasil analisis, ESP32-CAM akan mengklasifikasikan gambar sebagai fokus atau tidak fokus, tergantung pada nilai variansinya.



Gambar V.10 ESP32-CAM

Setelah Edge Impulse menghasilkan kode program untuk deteksi objek, langkah selanjutnya adalah mengimplementasikan sistem penerimaan dan penyimpanan data hasil deteksi tersebut di server. Pada program ini, server berbasis Flask digunakan untuk menerima gambar dan label yang dikirim dari ESP32. Gambar

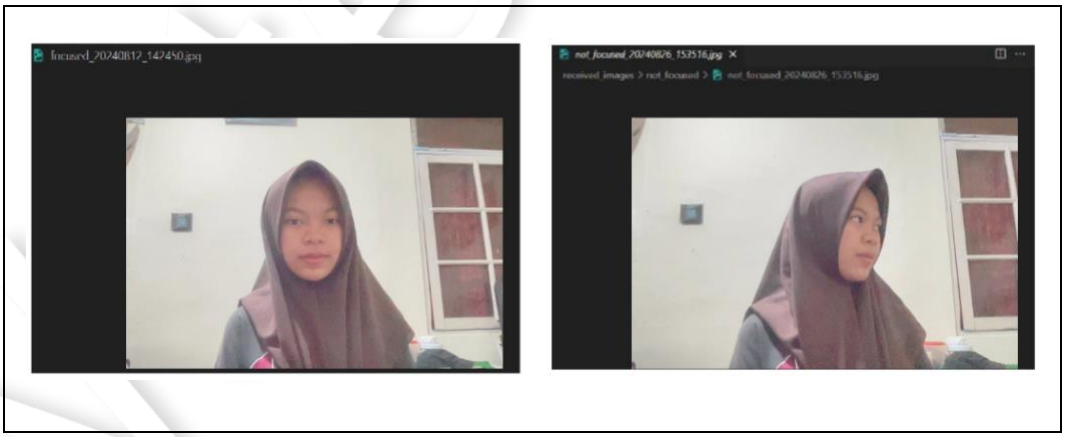
dikirim dalam format base64 dan dikodekan kembali menjadi format gambar menggunakan pustaka Python PIL (*Python Imaging Library*).

Setiap gambar yang diterima disertai dengan label deteksi, yaitu '*focused*' atau '*not focused*'. Berdasarkan label ini, gambar akan disimpan di direktori yang sesuai. Misalnya, gambar dengan label '*focused*' disimpan di direktori received_images/focused, sedangkan gambar dengan label '*not focused*' disimpan di direktori received_images/not_focused.

Penamaan file gambar dilakukan secara unik dengan menyertakan timestamp, sehingga setiap file yang disimpan memiliki nama yang berbeda dan mudah diidentifikasi. Dengan begitu, server dapat menerima dan menyimpan gambar-gambar hasil deteksi objek secara otomatis dan terorganisir dengan baik, yang nantinya dapat digunakan untuk analisis lebih lanjut. Berikut adalah kode programnya:

```
from flask import Flask, request, jsonify
from PIL import Image
import io
import base64
import os
from datetime import datetime

app = Flask(__name__)

# Directory to save images
SAVE_DIR = 'received_images'
FOCUSED_DIR = os.path.join(SAVE_DIR, 'focused')
NOT_FOCUSED_DIR = os.path.join(SAVE_DIR, 'not_focused')

# Create directories if they don't exist
for directory in [SAVE_DIR, FOCUSED_DIR, NOT_FOCUSED_DIR]:
    if not os.path.exists(directory):
        os.makedirs(directory)

@app.route('/upload', methods=['POST'])
def upload():
    try:
        data = request.json
        if 'image' not in data atau 'label' not in data:
            return jsonify({'error': 'Invalid data'}), 400

        # Decode base64 image
        image_data = base64.b64decode(data['image'])
        image = Image.open(io.BytesIO(image_data))
```

```
        # Create unique filename with timestamp and label
        label = data['label'].replace(" ", "_").lower()  #
Replace spaces in label with underscores and convert to
lower case
        timestamp =
datetime.now().strftime("%Y%m%d_%H%M%S")

        print(f"Received label: {label}")  # Print label
for debugging

        # Determine directory based on label
        if label == 'focused':
            dir_path = FOCUSED_DIR
        elif label == 'not_focused':
            dir_path = NOT_FOCUSED_DIR
        else:
            return jsonify({'error': 'Invalid label'}), 400

        image_filename = os.path.join(dir_path,
f"{label}_{timestamp}.jpg")
        image.save(image_filename)

        return jsonify({'message': 'Image and label
received successfully', 'predicted_label': label}), 200
    except Exception as e:
        print(f"Error: {e}")
        return jsonify({'error': 'Failed to process image
and label'}), 500

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```



Gambar V.11 Hasil percobaan klasifikasi objek fokus dan tidak fokus.

Gambar V.11 merupakan hasil dari percobaan klasifikasi objek fokus dan tidak fokus. penamaan file gambar dilakukan secara unik dengan menyertakan

timestamp, sehingga setiap file yang disimpan memiliki nama yang berbeda dan mudah diidentifikasi. Dengan begitu, server dapat menerima dan menyimpan gambar-gambar hasil deteksi objek secara otomatis dan terorganisir dengan baik, yang nantinya dapat digunakan untuk analisis lebih lanjut.

Bab VI Kesimpulan dan Saran

Pada Bab ini, menjelasakan kesimpulan dan saran dalam penelitian pemanfaatan komputasi tepi berbasis kecerdasan buatan

VI.1 Kesimpulan

Penelitian ini telah berhasil mengembangkan sistem deteksi kefokusan audien berdasarkan citra wajah yang ditangkap dan diproses langsung menggunakan peralatan kamera ESP32-CAM. Sistem ini mengintegrasikan konsep *edge computing* dan *machine learning*, di mana hasil pemrosesan gambar disimpan pada server Raspberry Pi, sehingga komputasi dilakukan di jaringan tepi (*edge*) untuk mendukung penghematan penggunaan sumber daya media transmisi. Metodologi yang digunakan dalam penelitian ini adalah CRISP-DM (*Cross-Industry Standard Process for Data Mining*), yang mencakup tahapan pemahaman bisnis, pemahaman data, persiapan data, pemodelan, evaluasi, dan *deployment*. Model kecerdasan buatan yang dibangun menggunakan arsitektur MobileNetV2 dan dioptimalkan dengan teknik augmentasi data. Hasil evaluasi menunjukkan bahwa model ini mencapai akurasi terbaik sebesar 71,43%, 69% untuk nilai AUC, 75% untuk nilai presisi, 76% untuk nilai recall, dan 76% untuk nilai F1 score. Meskipun akurasi ini masih dapat ditingkatkan, hasil ini menunjukkan potensi besar dalam penerapan machine learning untuk mendeteksi tingkat fokus audiens, terutama dalam konteks kegiatan belajar mengajar (KBM).

VI.2 Saran

Untuk penelitian selanjutnya dalam klasifikasi fokus dan tidak fokus seseorang berdasarkan pose kepala, beberapa saran berikut dapat dipertimbangkan.

1. Keragaman dataset perlu dilakukan dengan mengumpulkan data gambar yang beragam mengenai fokus dan tidak fokusnya seseorang guna meningkatkan representasi dan generalisasi model.
2. Peningkatan model dapat dilakukan dengan menggabungkan berbagai algoritma deep learning yang lebih kompleks untuk memanfaatkan model yang sudah dilatih pada dataset yang lebih besar dan beragam.

3. Implementasi pada perangkat edge seperti ESP32-CAM perlu terus dioptimalkan, termasuk pengembangan algoritma yang lebih efisien dan penggunaan hardware tambahan jika diperlukan untuk meningkatkan kinerja sistem.

DAFTAR PUSTAKA

Ab Wahab, M. N., Nazir, A., Ren, A. T. Z., Noor, M. H. M., Akbar, M. F., & Mohamed, A. S. A. (2021). Efficientnet-Lite and Hybrid CNN-KNN Implementation for Facial Expression Recognition on Raspberry Pi. *IEEE Access*, *9*, 134065–134080. https://doi.org/10.1109/ACCESS.2021.3113337

Afroze, S., & Hoque, M. M. (2020). *Classification of Attentional Focus based on Head Pose in Multi-Object Scenario*.

Afroze, S., Hossain, M. R., & Hoque, M. M. (2022). DeepFocus: A visual focus of attention detection framework using deep learning in multi-object scenarios. *Journal of King Saud University - Computer and Information Sciences*, *34*(10), 10109–10124. https://doi.org/10.1016/j.jksuci.2022.10.009

Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, *54*(15), 2787–2805. https://doi.org/10.1016/j.comnet.2010.05.010

Bai, C., Kumar, S., Leskovec, J., Metzger, M., Nunamaker, J. F., & Subrahmanian, V. S. (2019). *Predicting the Visual Focus of Attention in Multi-Person Discussion Videos*. https://en.wikipedia.org/wiki/The

BIRLOG, I.-A., BORCAN, D.-M., & COVRIG, G.-M. (2020). Internet of Things Hardware and Software. *Informatica Economica*, *24*(2/2020), 54–65. https://doi.org/10.24818/issn14531305/24.2.2020.05

Blessing, L. T. M., & Chakrabarti, A. (2009). DRM, a design research methodology. In *DRM, a Design Research Methodology*. https://doi.org/10.1007/978-1-84882-587-1

Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the internet of things. *MCC'12 - Proceedings of the 1st ACM Mobile Cloud Computing Workshop*, 13–15. https://doi.org/10.1145/2342509.2342513

Cao, K., Liu, Y., Meng, G., & Sun, Q. (2020). An Overview on Edge Computing Research. *IEEE Access*, *8*, 85714–85728.

https://doi.org/10.1109/ACCESS.2020.2991734

Caprolu, M., Di Pietro, R., Lombardi, F., & Raponi, S. (2019). Edge Computing Perspectives: Architectures, Technologies, and Open Security Issues. *Proceedings - 2019 IEEE International Conference on Edge Computing, EDGE 2019 - Part of the 2019 IEEE World Congress on Services*, 116–123. https://doi.org/10.1109/EDGE.2019.00035

Chakraborty, P., Yousuf, M. A., & Rahman, S. (2021). Predicting level of visual focus of human's attention using machine learning approaches. *Advances in Intelligent Systems and Computing*, *1309*, 683–694. https://doi.org/10.1007/978-981-33-4673-4_56

Chen, M., Challita, U., Saad, W., Yin, C., & Debbah, M. (2019). Artificial Neural Networks-Based Machine Learning for Wireless Networks: A Tutorial. *IEEE Communications Surveys and Tutorials*, *21*(4), 3039–3071. https://doi.org/10.1109/COMST.2019.2926625

Emambocus, B. A. S., Jasser, M. B., & Amphawan, A. (2023). A Survey on the Optimization of Artificial Neural Networks Using Swarm Intelligence Algorithms. *IEEE Access*, *11*(November 2022), 1280–1294. https://doi.org/10.1109/ACCESS.2022.3233596

ESF. (2021). *Edge Computing Kinerja*. https://esf.eurotech.com/v5.0.0/docs/edge-computing-kinerja

Gabbai, A. (2015). *Kevin Ashton Describes "the Internet of Things."* Smithsonian Magazine. https://www.smithsonianmag.com/innovation/kevin-ashton-describes-the-internet-of-things-180953749/

Gianto, L., Rukmana, T. H., Surya, S. I., & Bandung, Y. (2023). Automatic Early Detection System of Feverish Infectious Diseases in Indoor Public Areas. *Proceedings of the International Conference on Electrical Engineering and Informatics*, 1–6. https://doi.org/10.1109/ICEEI59426.2023.10346658

Goldberg, P., Sümer, Ö., Stürmer, K., Wagner, W., Göllner, R., Gerjets, P., Kasneci, E., & Trautwein, U. (2021). Attentive or Not? Toward a Machine

Learning Approach to Assessing Students' Visible Engagement in Classroom Instruction. In *Educational Psychology Review* (Vol. 33, Issue 1, pp. 27–49). Springer. https://doi.org/10.1007/s10648-019-09514-z

Ibnu Daqiqil. (2021). MACHINE LEARNING: Teori, Studi Kasus dan Implementasi Menggunakan Python. In *Badan Penerbit Universitas Riau* (Vol. 01).

ITU-T Y.2060, R. (2018). An overview of internet of things. *Journal of Advanced Research in Dynamical and Control Systems*, *10*(9), 659–665. https://doi.org/10.1201/9781003020905-1

Jebamani, S. A. (2021). *A Survey of Edge Computing in IOT devices*. https://ssrn.com/abstract=4023176

Jiang, B., Xu, W., Guo, C., Liu, W., & Cheng, W. (2019). *A Classroom Concentration Model Based on Computer Vision*.

Jovovic, I., Babic, D., Cakic, S., Popovic, T., Krco, S., & Knezevic, P. (2022). Face Mask Detection Based on Machine Learning and Edge Computing. *2022 21st International Symposium INFOTEH-JAHORINA, INFOTEH 2022 - Proceedings*, *951732*, 16–18. https://doi.org/10.1109/INFOTEH53737.2022.9751311

Kamath, V., Renuka, A., Kini, V. G., & Prabhu, S. (2024). Exploratory Data Preparation and Model Training Process for Raspberry Pi-Based Object Detection Model Deployments. *IEEE Access*, *12*(March), 45423–45441. https://doi.org/10.1109/ACCESS.2024.3381798

Ke, J., Zeng, S. Y., & Wang, J. J. (2022). Research on Image Caption Algorithm Based on Improved Attention Mechanism. *ISCTT 2021 - 6th International Conference on Information Science, Computer Technology and Transportation*, 108–112. https://doi.org/10.1109/ICSP62122.2024.10743367

Kong, Y., & Li, W. (2017). Research on recognition method of learning concentration based on face feature. *2017 IEEE International Conference on Cybernetics and Intelligent Systems, CIS 2017 and IEEE Conference on*

Robotics, Automation and Mechatronics, RAM 2017 - Proceedings, 2018-Janua, 334–338.

Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2022). A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

Liu, F., Tang, G., Li, Y., Cai, Z., Zhang, X., & Zhou, T. (2019a). A Survey on Edge Computing Systems and Tools. *Proceedings of the IEEE*, *107*(8), 1537–1562. https://doi.org/10.1109/JPROC.2019.2920341

Liu, F., Tang, G., Li, Y., Cai, Z., Zhang, X., & Zhou, T. (2019b). A Survey on Edge Computing Systems and Tools. *Proceedings of the IEEE*, *107*(8). https://doi.org/10.1109/JPROC.2019.2920341

Liu, T., Wang, J., Yang, B., & Wang, X. (2021). NGDNet: Nonuniform Gaussian-label distribution learning for infrared head pose estimation and on-task behavior understanding in the classroom. *Neurocomputing*, *436*, 210–220. https://doi.org/10.1016/j.neucom.2020.12.090

Ma, S., Zhang, X., Jia, C., Zhao, Z., Wang, S., & Wang, S. (2020). Image and Video Compression with Neural Networks: A Review. *IEEE Transactions on Circuits and Systems for Video Technology*, *30*(6), 1683–1698. https://doi.org/10.1109/TCSVT.2019.2910119

Manjunath, T. N., Pushpa, S. K., Hegadi, R. S., & Yogish, D. (2021). A Study on Edge Computing through Machine Learning for IoT Devices. *2021 International Conference on Forensics, Analytics, Big Data, Security, FABS 2021*, *1*, 1–6. https://doi.org/10.1109/FABS52071.2021.9702557

Massé, B., Ba, S., & Horaud, R. (2017). *Tracking Gaze and Visual Focus of Attention of People Involved in Social Interaction*. https://doi.org/10.1109/TPAMI.2017.2782819

Mujiyanto, Setyanto, A., Utami, E., & Kusrini, K. (2024). Facial Expression Recognition with Deep Learning and Attention Mechanisms: A Systematic

Review. *Proceedings - International Conference on Informatics and Computational Sciences*, 12–17. https://doi.org/10.1109/ICICoS62600.2024.10636857

Ozkan, Z., Bayhan, E., Namdar, M., & Basgumus, A. (2021). Object Detection and Recognition of Unmanned Aerial Vehicles Using Raspberry Pi Kinerja. *ISMSIT 2021 - 5th International Symposium on Multidisciplinary Studies and Innovative Technologies, Proceedings*, 467–472. https://doi.org/10.1109/ISMSIT52890.2021.9604698

Pan, J., & McElhannon, J. (2018). Future Edge Cloud and Edge Computing for Internet of Things Applications. *IEEE Internet of Things Journal*, *5*(1), 439–449. https://doi.org/10.1109/JIOT.2017.2767608

Patil, N., Ambatkar, S., & Kakde, S. (2017). IoT based smart surveillance security system using raspberry Pi. *Proceedings of the 2017 IEEE International Conference on Communication and Signal Processing, ICCSP 2017*, *2018-Janua*, 344–348. https://doi.org/10.1109/ICCSP.2017.8286374

Premsankar, G., Di Francesco, M., & Taleb, T. (2018). Edge Computing for the Internet of Things: A Case Study. *IEEE Internet of Things Journal*, *5*(2), 1275–1284. https://doi.org/10.1109/JIOT.2018.2805263

Rosero-Montalvo, P. D., Tozun, P., & Hernandez, W. (2024). Optimized CNN Architectures Benchmarking in Hardware-Constrained Edge Devices in IoT Environments. *IEEE Internet of Things Journal*, *11*(11), 20357–20366. https://doi.org/10.1109/JIOT.2024.3369607

Samie, F., Bauer, L., & Henkel, J. (2019). From cloud down to things: An overview of machine learning in internet of things. *IEEE Internet of Things Journal*, *6*(3), 4921–4934. https://doi.org/10.1109/JIOT.2019.2893866

Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal*, *3*(5), 637–646. https://doi.org/10.1109/JIOT.2016.2579198

Singh, T., Mohadikar, M., Gite, S., Patil, S., Pradhan, B., & Alamri, A. (2021).

Attention Span Prediction Using Head-Pose Estimation with Deep Neural Networks. *IEEE Access*, *9*, 142632–142643.

Slamet Purwo Santoso, & Fajar Wijayanto. (2022). Rancang Bangun Akses Pintu Dengan Sensor suhu Dan Handsanitizer Otomatis Berbasis Arduino. *Jurnal Elektro*, *10*(1), 20–31. https://jurnalteknik.unkris.ac.id/index.php/jie/article/view/137/134

Tedjopurnomo, D. A., Bao, Z., Zheng, B., Choudhury, F. M., & Qin, A. K. (2022). A Survey on Modern Deep Neural Network for Traffic Prediction: Trends, Methods and Challenges. *IEEE Transactions on Knowledge and Data Engineering*, *34*(4), 1544–1561. https://doi.org/10.1109/TKDE.2020.3001195

Vaishnavi, C., & Palaniswamy, S. (2022). Emotion Recognition From Online Classroom Videos Using Meta Learning. *ASSIC 2022 - Proceedings: International Conference on Advancements in Smart, Secure and Intelligent Computing*. https://doi.org/10.1109/ASSIC55218.2022.10088292

Wang, F., Zhang, M., Wang, X., Ma, X., & Liu, J. (2020). Deep Learning for Edge Computing Applications: A State-of-the-Art Survey. *IEEE Access*, *8*, 58322–58336. https://doi.org/10.1109/ACCESS.2020.2982411

Wicaksono, M. A., & Bandung, Y. (2022). CoAP Evaluation for Node-Edge Communication on Wireless Multimedia Sensor Network. *2022 International Conference on Information Technology Systems and Innovation, ICITSI 2022 - Proceedings*, 122–127. https://doi.org/10.1109/ICITSI56531.2022.9970953

Xuanhao, Q., & Min, Z. (2023). A Review of Attention Mechanisms in Computer Vision. *2023 8th International Conference on Image, Vision and Computing, ICIVC 2023*, 577–583. https://doi.org/10.1109/ICIVC58118.2023.10270435

Yingsaeree, C., Treleaven, P., & Centre, U. K. (2010). *Cloud Computing For Mobile Users: Can Offloading Computation Save Energy*. *April*, 36–43.

Zareen, S. (2019). Artificial Intelligence / Machine Learning in IoT for Authentication and Authorization of Edge Devices. *International Conference on Applied and Engineering Mathematics (ICAEM)*, 220–224.

Zeng, H., Shu, X., Wang, Y., Wang, Y., Zhang, L., Pong, T. C., & Qu, H. (2021). EmotionCues: Emotion-Oriented Visual Summarization of Classroom Videos. *IEEE Transactions on Visualization and Computer Graphics*, *27*(7), 3168–3181. https://doi.org/10.1109/TVCG.2019.2963659

Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., & Parmar, M. (2024). A review of convolutional neural networks in computer vision. In *Artificial Intelligence Review* (Vol. 57, Issue 4). Springer Netherlands. https://doi.org/10.1007/s10462-024-10721-6

Ziliwu, J. R., Setyawan, G. C., & Budiati, H. (2024). Penerapan ESP32-CAM dan TinyML dalam Klasifikasi Gambar Buah dan Sayuran. *Jutisi : Jurnal Ilmiah Teknik Informatika Dan Sistem Informasi*, *13*(1), 584. https://doi.org/10.35889/jutisi.v13i1.1869