

# **SKRIPSI**

## **SISTEM WEARABLE PENDETEKSI JATUH DENGAN IMPLEMENTASI EDGE COMPUTING**

**Disusun dan diajukan oleh:**

**SAPHIRA NOER S.  
D121 17 1520**



**PROGRAM STUDI SARJANA TEKNIK INFORMATIKA  
FAKULTAS TEKNIK  
UNIVERSITAS HASANUDDIN  
GOWA  
2024**

## LEMBAR PENGESAHAN SKRIPSI

### SISTEM WEARABLE PENDETEKSI JATUH DENGAN IMPLEMENTASI EDGE COMPUTING

Disusun dan diajukan oleh

**SAPHIRA NOER S.**  
**D121 17 1520**

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian  
Studi Program Sarjana Program Studi Teknik Informatika  
Fakultas Teknik Universitas Hasanuddin  
Pada tanggal 01 Juli 2024  
dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui,

Pembimbing Utama,

Pembimbing Pendamping,

  
Dr. Adnan, S.T., M.T.

NIP 197404262005011002

  
Dr. Ir. Zahir Zainuddin, M.Sc.

NIP 196404271989101002

  
Ketua Program Studi,

Prof. Dr. Ir. Indrabayu, ST., MT., M.Bus.Sys., IPM, ASEAN. Eng.

NIP 197507162002121004

## PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini;

Nama : Saphira Noer S.  
NIM : D121171520  
Program Studi : Teknik Informatika  
Jenjang : S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

Sistem Wearable Pendeteksi Jatuh Dengan Implementasi Edge Computing

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain dan bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Semua informasi yang ditulis dalam skripsi yang berasal dari penulis lain telah diberi penghargaan, yakni dengan mengutip sumber dan tahun penerbitannya. Oleh karena itu semua tulisan dalam skripsi ini sepenuhnya menjadi tanggung jawab penulis. Apabila ada pihak manapun yang merasa ada kesamaan judul dan atau hasil temuan dalam skripsi ini, maka penulis siap untuk diklarifikasi dan mempertanggungjawabkan segala resiko.

Segala data dan informasi yang diperoleh selama proses pembuatan skripsi, yang akan dipublikasi oleh Penulis di masa depan harus mendapat persetujuan dari Dosen Pembimbing.

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

Gowa, 10 Juli 2024

Yang Menyatakan



SAPHIRA NOER S.

## ABSTRAK

**SAPHIRA NOER S.** Sistem Wearable Pendeteksi Jatuh Dengan Implementasi Edge Computing (dibimbing oleh Adnan dan Zahir Zainuddin)

Meningkatnya kebutuhan akan sistem deteksi jatuh yang efektif dan akurat pada dekade terakhir ini meningkat terutama bagi populasi lansia yang rentan terhadap cedera akibat jatuh. Teknologi sensor akselerometer telah berkembang pesat dan menawarkan solusi potensial untuk mendeteksi jatuh secara *real-time* sehingga sistem wearable banyak diimplementasikan untuk sistem deteksi jatuh. Namun, tantangan utama adalah *deployment* model *deep learning* yang dapat membedakan antara jatuh dan aktivitas sehari-hari dengan tingkat akurasi tinggi ke dalam perangkat bersumber-daya terbatas.

Penelitian ini bertujuan untuk mengembangkan sistem deteksi jatuh mengimplementasikan *edge computing* dengan dukungan Edge Impulse untuk membangun model *deep learning* dan Bluetooth Low Energy untuk memungkinkan komunikasi antar-perangkat berdaya rendah dengan konsumsi daya yang minim. Edge computing diharapkan dapat mengefisienkan pemrosesan dan pengiriman data dengan cara melakukan pemrosesan lebih dekat secara fisik dengan sumber data. Tujuan lain dari penelitian ini adalah untuk mengevaluasi kinerja model yang telah disematkan ke dalam perangkat edge bila dibandingkan dengan akurasi model secara teori.

Data akselerasi tri-aksial UniMiB SHAR yang mengandung data akselerasi jatuh dan kegiatan sehari-hari diproses menggunakan filter Butterworth orde keenam dengan frekuensi *cut-off* 3 Hz untuk menghilangkan *noise*. Selanjutnya, data yang telah difilter dihitung besaran magnitudonya dan digunakan sebagai input untuk model *deep learning*. Perangkat edge dan wearable yang masing-masing merupakan mikrokontroler saling berkomunikasi menggunakan Bluetooth Low Energy, dimana wearable akan mengirimkan data akselerasi kepada edge yang akan memproses data tersebut untuk menghasilkan keputusan.

Hasil penelitian menunjukkan bahwa model deteksi jatuh yang dikembangkan memiliki tingkat akurasi yang tinggi dalam membedakan jatuh dari kegiatan sehari-hari, yaitu sebesar 91%.

Kata Kunci: Edge Computing, Jatuh, Wearable

## ABSTRACT

**SAPHIRA NOER S.** *Fall Detection Wearable System with Edge Computing*  
(supervised by Adnan and Zahir Zainuddin)

The increasing need for effective and accurate fall detection systems in the last decade has increased especially for the elderly population who are vulnerable to fall injuries. Accelerometer sensor technology has developed rapidly and offers a potential solution to detect falls in real-time, thus wearable systems are widely implemented for fall detection systems. However, a major challenge is the deployment of deep learning models that can distinguish between falls and daily activities with a high degree of accuracy into limited-resource devices.

This research aims to develop a fall detection system implementing edge computing with the support of Edge Impulse to build deep learning models and Bluetooth Low Energy to enable low-power inter-device communication with minimal power consumption. Edge computing is expected to streamline data processing and transmission by performing processing physically closer to the data source. Another objective of this research is to evaluate the performance of the deep learning model that have been embedded into the edge device when compared to the acquired theoretical accuracy of said model.

The UniMiB SHAR dataset that contains tri-axial acceleration data of falls and daily activities were processed using a sixth-order Butterworth filter with a cut-off frequency of 3 Hz to remove noise. Next, the filtered data is calculated for magnitude and used as input for the deep learning model. The edge device and wearable, each of which is a microcontroller, communicate with each other using Bluetooth Low Energy, where the wearable will send acceleration data to the edge which will process the data to make a decision.

The results show that the fall detection model developed has a high accuracy rate in distinguishing falls from daily activities, which is 91%.

Keywords: Edge Computing, Fall, Wearable

## DAFTAR ISI

LEMBAR PENGESAHAN SKRIPSI .....	<b>Error! Bookmark not defined.</b>
PERNYATAAN KEASLIAN .....	<b>Error! Bookmark not defined.</b>
ABSTRAK .....	iv
ABSTRACT .....	v
DAFTAR ISI .....	vi
DAFTAR GAMBAR .....	vii
DAFTAR TABEL .....	ix
DAFTAR SINGKATAN DAN ARTI SIMBOL .....	x
DAFTAR LAMPIRAN .....	xi
KATA PENGANTAR .....	xii
BAB I PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	4
1.3 Tujuan Penelitian .....	4
1.4 Manfaat Penelitian .....	4
1.5 Ruang Lingkup .....	5
BAB II TINJAUAN PUSTAKA .....	6
2.1 Jatuh .....	6
2.2 Internet of Things .....	6
2.3 Edge Computing .....	7
2.4 TinyML .....	8
2.5 Edge Impulse .....	9
2.6 Bluetooth Low Energy .....	10
2.7 Arduino Nano 33 BLE Sense .....	12
2.8 Signal Vector Magnitude / Sum Vector Magnitude (SMV) .....	12
2.9 Filter .....	14
2.10 UniMiB SHAR .....	16
2.11 Deep Neural Network (DNN) .....	16
BAB 3 METODE PENELITIAN/PERANCANGAN .....	20
3.1 Tahapan Penelitian .....	20
3.2 Waktu dan Lokasi Penelitian .....	20
3.3 Instrumen Penelitian .....	20
3.4 Rancangan Sistem .....	21
3.5 Evaluasi Model .....	40
3.6 Evaluasi Sistem .....	41
BAB 4 HASIL DAN PEMBAHASAN .....	42
4.1 Hasil Rancangan Sistem .....	42
4.2 Evaluasi Model .....	45
4.3 Evaluasi Sistem .....	51
BAB 5 KESIMPULAN DAN SARAN .....	54
5.1 Kesimpulan .....	54
5.2 Saran .....	54
DAFTAR PUSTAKA .....	55

## DAFTAR GAMBAR

Gambar 1 Ilustrasi perbedaan <i>cloud computing</i> dengan <i>edge computing</i> ( <i>Edge Computing vs. Cloud Computing: What's the Difference?</i> , 2021).....	8
Gambar 2 Diagram venn korelasi ML, IoT, dan TinyML (Tomas, 2021).....	8
Gambar 3 Arduino Nano 33 BLE Sense (Machine Learning on Arduino Nano 33 BLE Sense, 2021) .....	12
Gambar 4 Ilustrasi akselerasi tiga dimensi (Alushi, 2022) .....	14
Gambar 5 Jenis-jenis filter (Storr, 2013) .....	15
Gambar 6 Diagram tahapan penelitian.....	20
Gambar 7 Diagram blok sistem .....	21
Gambar 8 Alur kerja (a) perangkat edge dan (b) perangkat wearable .....	22
Gambar 9 Letak pemakaian <i>wearable</i> .....	23
Gambar 10 (a) EDA wearable dan (b) desain fritzing wearable .....	24
Gambar 11 Alur pembangunan dan pengujian model .....	25
Gambar 12 Isi salah satu sub-array .....	26
Gambar 13 Visualisasi data UniMIB SHAR.....	26
Gambar 14 Alur kerja program untuk konversi dataset .....	27
Gambar 15 Hasil konversi dataset: (a) isi file CSV dan (b) format nama file setelah konversi. ....	28
Gambar 16 Opsi pada 'Edge Impulse Studio Uploader' .....	29
Gambar 17 Laman 'Data Acquisition' .....	30
Gambar 18 Tampilan laman "Create Impulse" .....	31
Gambar 19 Pilihan pada (a) blok pemrosesan dan (b) blok pembelajaran .....	31
Gambar 20 Tampilan laman "Impulse Design" .....	32
Gambar 21 Pengaturan <i>Spectral Features</i> .....	33
Gambar 22 Hasil ekstraksi fitur.....	34
Gambar 23 Pengaturan <i>Classifier</i> .....	35
Gambar 24 Pengaturan <i>Deployment</i> .....	36
Gambar 25 Isi file ZIP <i>deployment</i> model edge impulse .....	36
Gambar 26 <i>Flowchart</i> sistem edge .....	37
Gambar 27 <i>Flowchart</i> sistem wearable .....	37
Gambar 28 <i>Source code</i> konektivitas BLE edge.....	38
Gambar 29 <i>Source code</i> konektivitas BLE wearable .....	39
Gambar 30 Tampak rangkaian <i>wearable</i> (a) di luar <i>casing</i> , (b) di dalam <i>casing</i> tanpa tutup, dan (c) di dalam <i>casing</i> dengan tutup. ....	42
Gambar 31 Perhubungan edge.....	43
Gambar 32 Perhubungan wearable .....	43
Gambar 33 Nilai-nilai akselerasi beserta magnitudonya .....	44
Gambar 34 Notifikasi jatuh dari edge .....	44
Gambar 35 Hasil observasi <i>threshold</i> model .....	45
Gambar 36 Keluaran <i>command prompt</i> ketika terdeteksi adanya jatuh .....	45
Gambar 37 Pesan Whatsapp yang dikirimkan .....	45
Gambar 38 Hasil pelatihan model .....	46
Gambar 39 Akselerasi ketika berlari ringan.....	47
Gambar 40 Keputusan model untuk kegiatan berlari .....	47
Gambar 41 Akselerasi ketika melakukan lompatan beruntun.....	48

Gambar 42 Keputusan model untuk serangkaian lompatan.....	48
Gambar 43 Akselerasi ketika jatuh.....	49
Gambar 44 Keputusan model ketika jatuh .....	49
Gambar 45 Keluaran <i>serial monitor</i> wearable dengan <i>timestamp</i> .....	52
Gambar 46 Keluaran <i>serial monitor</i> edge dengan <i>timestamp</i> .....	53
Gambar 47 Keluaran <i>command prompt</i> yang menunjukkan total waktu yang dibutuhkan untuk mengirimkan pesan Whatsapp.....	53
Gambar 48 Pesan yang dikirimkan pada 08:51 mengenai jatuh yang terjadi pada pukul 08.50 .....	53



## DAFTAR TABEL

Tabel 1 <i>Confusion matrix</i> .....	19
Tabel 2 Spesifikasi subjek pengujian.....	40
Tabel 3 Skema pengujian sistem .....	41
Tabel 4 Hasil pengujian pada subjek 1 (155 cm) .....	50
Tabel 5 Hasil pengujian pada subjek 2 (160 cm) .....	50
Tabel 6 Hasil pengujian pada subjek 3 (170 cm) .....	50
Tabel 7 Hasil pengujian performa sistem.....	51
Tabel 8 Cuplikan interval keluaran serial monitor wearable .....	52
Tabel 9 Cuplikan interval keluaran serial monitor edge .....	53

## DAFTAR SINGKATAN DAN ARTI SIMBOL

Lambang/Singkatan	Arti dan Keterangan
IoT	Internet of Things
AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning
ANN	Artificial Neural Network
DNN	Deep Neural Network
MLOps	Machine Learning Operations
HAR	Human Activity Recognition
ADL	Activities Of Daily Living
BLE	Bluetooth Low Energy
RFID	Radio Frequency Identification
SBC	Single Board Computer
TFLM	Tensor Flow
UART	Universal Asynchronous Receiver Transmitter
UUID	Universally Unique Identifier
RAM	Random Access Memory
EON	Edge Optimized Neural
SMV	Sum Vector Magnitude / Signal Vector Magnitude
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
CSV	Comma Separated Values
DSP	Digital Signal Processing
FFT	Fast Fourier Transform

## DAFTAR LAMPIRAN

Lampiran 1. <i>Source code</i> .....	59
Lampiran 2. Interval tiap keluaran serial monitor wearable (Sampel 5 detik) .....	59
Lampiran 3. Interval tiap keluaran serial monitor edge (Sampel 30 detik) .....	60
Lampiran 4. Lembar Perbaikan Skripsi .....	61

## KATA PENGANTAR

Bismillahirrahmanirrahim. Alhamdulillah rabbil ‘alamin. Skripsi berjudul “Sistem Wearable Pendeteksi Jatuh Dengan Implementasi Edge Computing” ini mengambil permasalahan fatalitas jatuh yang dapat dimitigasi dengan cara mengimplementasikan *edge computing* dalam sebuah sistem deteksi jatuh. Skripsi yang menjadi syarat kelulusan jenjang Strata-1 di Universitas Hasanuddin ini tidak akan dapat diselesaikan jikalau bukan atas kehendak Allah Subhanahu wa Ta’ala dan bantuan setiap orang-orang yang telah Allah pertemukan atau sampaikan jasanya kepada penulis:

1. Keluarga penulis, Hafizhahumullah.
2. Para Pembimbing Skripsi penulis; Bapak Adnan, ST., MT., Ph.D. serta Bapak Dr. Ir. Zahir Zainuddin, M.Sc. yang telah membimbing penulis dalam pengerjaan skripsi dengan sabar.
3. Bapak Prof. Dr. Ir. Indrabayu, ST., MT, M.Bus.sys. selaku Pembimbing Akademik penulis dan Kepala Departemen Teknik Informatika yang telah memberi banyak motivasi dan inspirasi kepada penulis selama studi.
4. Segenap staf kampus, terkhusus para staf gedung elektro dan COT yang senantiasa berbagi dengan penulis baik berupa makanan maupun pengalaman tangan pertama yang begitu berharga.
5. Rekan-rekan Informatika terkhusus Kak Fadel, Amir, Fadhil, Ikbil, Devy, Irma, Nurina, Yudi, Alisyah, Herlina, Wahyu, dan Kak Andy yang telah banyak membantu dan menemani penulis selama mengerjakan skripsi.
6. Orang-orang yang belum pernah ditemui oleh penulis yang jerih payahnya tersampaikan dalam bentuk ide, fasilitas, maupun doa.
7. My past selves, for not giving up on me.

Semoga Allah membalas dengan sebaik-baiknya kebaikan. Terima kasih.

Gowa, Juli 2024  
Penulis,

Saphira Noer S.

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Secara global, jatuh merupakan penyebab utama kedua untuk kematian akibat cedera yang tidak disengaja. Setiap tahun diperkirakan sebanyak 684.000 individu meninggal karena jatuh. Definisi umum jatuh yaitu kejadian yang mengakibatkan seseorang berbaring secara tidak sengaja di permukaan yang lebih rendah. Sebanyak 37,3 juta kejadian jatuh yang cukup parah memerlukan perhatian medis setiap tahunnya. Secara global, kejadian jatuh bertanggung jawab untuk lebih dari 38 juta DALY (usia kematian di bawah angka harapan hidup dengan dampak dari cacat) yang bertambah tiap tahun (*Falls*, 2021).

Diantara upaya mitigasi jatuh yang dapat dilakukan adalah dengan mendeteksi jatuh sesaat setelah jatuh itu terjadi dan memberitahukan pihak terkait. Beberapa jenis pendeteksi jatuh yaitu dengan konsep sistem *context-aware* dan sistem *wearable*. Deteksi jatuh yang menggunakan konsep *context-aware* dapat diterapkan dengan cara menggunakan kamera dan sensor-sensor *ambient* lainnya seperti kamera dalam ruangan, sensor tekanan pada lantai, atau mikrofon di sekitar pengguna yang ingin dipantau. Namun, menurut Gold dan Shaw, mobilitas pasien dapat dipantau secara lebih adaptif dan praktis menggunakan sensor-sensor ringan yang dapat dikenakan secara langsung oleh pasien karena meluasnya penggunaan perangkat elektronik menyebabkan penurunan biaya produksi sehingga sistem deteksi jatuh dengan penerapan sistem *wearable* semakin efisien (Gold & Shaw, 2022). Implementasi paling umum saat ini berupa sensor miniatur yang dapat dikenakan pada tubuh untuk mengamati perubahan gaya hidup dan perilaku pemakainya. Sebagai hasilnya, penelitian ekstensif telah dilakukan dalam pengembangan algoritma penalaran untuk menyimpulkan aktivitas dari data sensor-sensor *wearable*. Akselerometer merupakan sensor yang paling sering digunakan untuk tujuan pengenalan aktivitas dan dapat menyediakan akurasi klasifikasi tinggi (Janidarmian et al., 2017).

Proses deteksi jatuh dapat dilakukan secara langsung di perangkat *wearable*, atau di *cloud*. Namun, alasan perangkat *wearable* banyak digunakan oleh

masyarakat adalah kepraktisan dan kenyamanan karena ukuran perangkat yang kecil. Batasan ukuran ini menyebabkan terbatasnya kapasitas penyimpanan dan kemampuan komputasi perangkat-perangkat *wearable*. Pendeteksian di *cloud* juga dapat dilakukan. Akan tetapi, memanfaatkan layanan *cloud computing* seringkali bukanlah opsi yang ekonomis, dan layanan yang disediakan secara gratis seringkali memiliki batasan. Selain itu, meningkatnya pertumbuhan lalu lintas *Big Data* yang diakibatkan oleh pertumbuhan pesat IoT (Internet of Things) mengakibatkan koneksi data dan komunikasi data melalui jaringan menjadi lambat. Solusi yang efektif untuk permasalahan ini yaitu *Edge computing*. *Edge computing* merupakan komputasi yang terjadi di dalam perangkat *edge* yang peletakkannya dekat dengan pusat data yang umumnya berupa perangkat-perangkat yang berinteraksi langsung dengan pengguna akhir (*end user*). Oleh karena itu, *edge* menyumbang latensi yang lebih rendah bila dibandingkan dengan *cloud*. Data-data yang diperoleh dari *edge* akan diproses atau disimpan secara lokal, lalu semua data yang telah diproses dikirim ke *cloud* sehingga mengurangi beban komputasi dan jaringan, serta menghindari potensi *bottleneck* di layanan pusat. (*Edge Computing*, 2019).

Pengimplementasian *wearable* untuk memantau atau mengawasi kesehatan merupakan salah satu *use case* IoT yang paling umum. IoT merujuk pada objek-objek fisik yang disematkan sensor-sensor dan aktuator-aktuator yang saling berkomunikasi dengan sistem-sistem komputasi melalui jaringan kabel maupun nirkabel untuk memantau atau mengatur kehidupan sehari-hari (*What Is the Internet of Things?*, 2022). Transmisi nirkabel antara sensor-sensor ini dan *base station* berperan penting. Terdapat banyak solusi transmisi nirkabel, seperti teknologi IEEE 802.11 (WiFi), ZigBee, Bluetooth *Classic*, dan Bluetooth Low Energy (BLE). Meskipun teknologi WiFi dapat mentransmisi data dengan kecepatan tinggi, WiFi membutuhkan konsumsi daya yang besar untuk memfasilitasi lalu lintas *throughput* yang tinggi. Selain itu, WiFi tidak praktis dari perspektif koneksi antar perangkat. BLE yang menggabungkan konsumsi daya rendah dan penggunaan meluas merupakan protokol komunikasi radio jarak dekat yang memiliki kemampuan untuk mengurangi konsumsi daya dan biaya perangkat pada perangkat-perangkat berdaya rendah (Liu et al., 2021).

Selama dekade terakhir ini, kita telah melihat pengembangan algoritma *Machine Learning* (ML) bersamaan dengan pengembangan *Internet of Things* di bidang mikroelektronik, komunikasi, dan teknologi informasi. Miliaran perangkat IoT terhubung ke internet; dengan jumlah perangkat IoT yang luas datang pula produksi massal platform IoT. Peran dari perangkat-perangkat ini adalah untuk mendeteksi fitur fisik dari lingkungan penempatannya—dua puluh empat jam sehari, tujuh hari seminggu. Hal ini menyebabkan peningkatan volume data yang dihasilkan yang sebaliknya juga memerlukan kinerja komputasi yang tinggi dan ruang penyimpanan yang besar. Dewasa ini, algoritma *Machine Learning* diintegrasikan dengan perangkat IoT agar mampu memproses jumlah data yang besar dan memungkinkan perangkat-perangkat tersebut untuk dapat mengambil keputusan. *Deep Neural Network* (DNN) atau *Deep Learning* (DL) yang kerap kali dimanfaatkan untuk mengolah data pada perangkat-perangkat *edge* sering dihadapkan dengan tantangan-tantangan berupa: terbatasnya kapabilitas perangkat-perangkat *edge* bila dibandingkan dengan besarnya model DL sehingga membebani biaya komputasi, sumber daya, daya, memori, dan waktu; serta tantangan lama waktu yang dibutuhkan bagi model untuk membuat keputusan. TinyML merupakan pendekatan yang dikembangkan untuk menghubungkan ML dengan perangkat *edge*. TinyML memungkinkan *deployment* model DL berukuran kecil ke dalam perangkat *edge* yang memiliki keterbatasan sumber daya yang signifikan. Dengan TinyML, analisis dan interpretasi data dapat dilakukan secara lokal pada perangkat sehingga pengambilan keputusan atau tindakan dapat dilakukan secara *real time* (Alajlan & Ibrahim, 2022).

Salah satu platform *Machine Learning Operations* (MLOps) yang memungkinkan pengguna individu maupun industri untuk merambah ke ranah TinyML adalah Edge Impulse, sebuah layanan berbasis *cloud* untuk pengembangan sistem-sistem *edge* dan tersemat. Edge Impulse menyederhanakan proses pengumpulan data, pelatihan model deep learning beserta evaluasinya, dan *deployment* model ke dalam perangkat-perangkat *edge computing*. Para pengguna dipermudah untuk berinteraksi dengan sistem dalam proses pelatihan model dan *deployment* oleh antarmuka grafis (GUI) and API. Selain itu, Edge Impulse menyediakan *library* C/C++ yang portable dan ekstensif guna meng-enkapsulasi

kode *preprocessing* dan model untuk menyederhanakan proses inferensi pada berbagai perangkat sekaligus mengoptimalkan waktu inferensi dan konsumsi memori model (Hymel et al., 2022).

Berdasarkan paparan di atas, penelitian ini bermaksud untuk menghasilkan sebuah sistem deteksi jatuh memanfaatkan sensor akselerometer yang terintegrasi dalam perangkat *wearable* untuk memantau aktivitas pengguna. Sebagian besar pemrosesan data dijalankan pada perangkat *edge* yang telah disematkan sebuah model *machine learning*, sedangkan *cloud* hanya digunakan untuk mengirim notifikasi ke kontak darurat yang telah ditentukan. Guna mengoptimalkan penggunaan daya pada *wearable*, protokol komunikasi yang digunakan adalah BLE.

## 1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas dapat disimpulkan rumusan masalah:

1. Bagaimana mengimplementasikan *edge computing* dalam membangun sistem deteksi jatuh?
2. Bagaimana performa model dalam mendeteksi jatuh?

## 1.3 Tujuan Penelitian

Tujuan dari penelitian yang dilakukan yaitu:

1. Membangun sistem deteksi jatuh mengimplementasikan *edge computing*.
2. Mengukur performa model dalam mendeteksi jatuh.

## 1.4 Manfaat Penelitian

Penelitian ini diharapkan dapat menambah pengetahuan pembaca dan menjadi referensi yang baik bagi penelitian-penelitian lain yang mengangkat tema *edge computing* dan IoT.



## 1.5 Ruang Lingkup

Ruang lingkup dalam membangun sistem ini adalah:

1. Penelitian ini hanya menggunakan dataset UniMIB SHAR sebagai data pelatihan.
2. Pelatihan model dilakukan di Edge Impulse.
3. Kategori pergerakan yang dideteksi dalam penelitian ini dikerucutkan menjadi dua kelas, yaitu ADL dan Fall.
4. Penelitian ini hanya mendeteksi jatuh berdasarkan sinyal akselerometer.
5. Perangkat *wearable* pada penelitian ini dikenakan di area pinggang atau paha (saku celana) pengguna berdasarkan konfigurasi yang dijabarkan pada bab-bab berikutnya.
6. Penelitian ini hanya menggunakan satu buah perangkat *wearable*.

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **2.1 Jatuh**

Menurut WHO, jatuh yang tidak disengaja merupakan penyebab kematian akibat cedera paling banyak kedua setelah kecelakaan transportasi. Setiap tahun, diperkirakan 684 ribu kasus jatuh mengakibatkan kematian di seluruh dunia, dan 37,3 juta kasus jatuh diperlukan perhatian medis darurat. Meskipun hampir 40% dari total disabilitas akibat jatuh di seluruh dunia terjadi pada anak-anak, pengukuran ini mungkin tidak secara akurat mencerminkan dampak disabilitas akibat jatuh pada lansia. Selain itu, secara statistik diestimasikan terjadi kematian setiap 19 menit disebabkan cedera akibat jatuh oleh lansia (*Falls*, 2021; Pugh et al., 2019). Jatuh terjadi sepanjang hidup manusia, tetapi menjadi lebih berisiko pada orang yang berusia lebih dari 50 tahun. Beberapa penelitian menunjukkan bahwa sekitar 30% lansia berusia lebih dari 65 tahun mengalami jatuh setiap tahunnya. Persentase ini meningkat menjadi 40% untuk orang yang berusia lebih dari 80 tahun (Lahouar et al., 2024).

#### **2.2 Internet of Things**

*Internet of Things* (IoT) merujuk pada objek-objek fisik yang disematkan sensor-sensor (untuk memantau perubahan lingkungan) dan aktuator-aktuator (yang menerima sinyal dari sensor dan melakukan sesuatu sebagai respon dari perubahan lingkungan yang terjadi) yang saling berkomunikasi dengan sistem-sistem komputasi melalui jaringan kabel maupun nirkabel untuk memantau atau mengendalikan dunia nyata secara digital. Objek-objek fisik tersebut tidak mesti benda buatan, melainkan dapat berupa objek-objek di alam seperti manusia dan hewan (*What Is the Internet of Things?*, 2022).

Konsep IoT diciptakan oleh anggota komunitas pengembangan *Radio Frequency Identification* (RFID) pada tahun 1999. Saat ini, perkembangan IoT sangat meningkat didorong oleh pertumbuhan perangkat seluler, komunikasi yang semakin mudah dilakukan, *cloud computing*, dan *data analytics*. Tujuan IoT adalah untuk memungkinkan segala sesuatunya selalu terhubung kapan saja dengan apa

saja dan siapa saja yang ideal menggunakan jaringan dan layanan apapun. IoT mengacu pada penggunaan perangkat dan sistem yang terhubung untuk memanfaatkan data yang diperoleh dari sensor, aktuator tersemat dalam mesin, atau objek fisik lainnya (Thamrin et al., 2020).

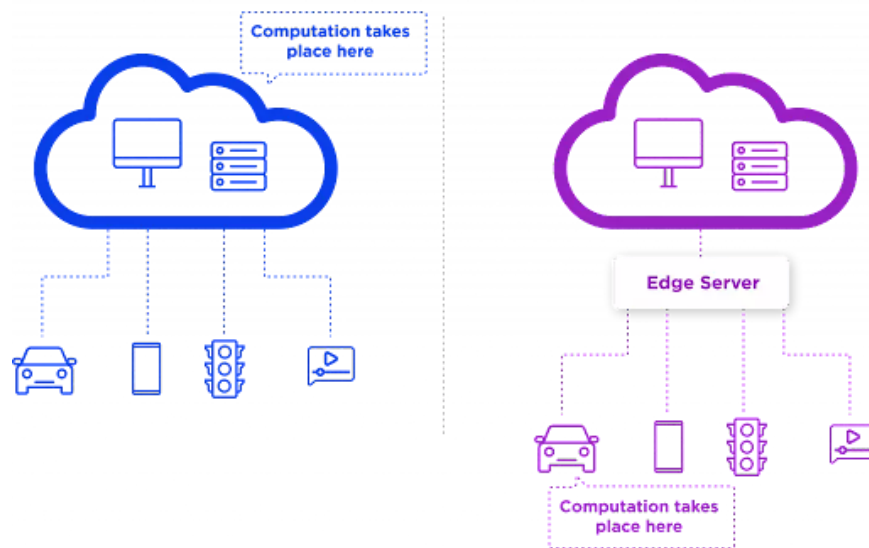
### 2.3 Edge Computing

*Cloud computing* adalah komputasi, penyimpanan, pelayanan, dan sumber daya aplikasi yang terpusat dan tervirtualisasi melalui Internet. *Cloud computing* memisahkan layanan dari infrastruktur asalnya sehingga mengeliminasi biaya dan kompleksitas mengelola infrastruktur teknologi informasi secara independen. Namun, *cloud computing* tidak mampu melayani aplikasi IoT secara *real-time* karena infrastruktur *cloud* biasanya terletak di beberapa tempat yang secara fisik jauh dari pengguna. Selain itu, kebutuhan aplikasi IoT diantara lain meliputi latensi rendah, *jitter* rendah, *bandwidth* tinggi, dan mobilitas tinggi (Mansouri & Babar, 2021).

Untuk memenuhi kebutuhan aplikasi IoT, dibutuhkan layanan berbasis *cloud* yang dekat dengan sumber data untuk memproses, menganalisis, dan memfilter data untuk mengambil keputusan yang tepat. Ekstensi layanan komputasi dari paradigma *cloud* ke tepi jaringan adalah *edge computing*. *Edge computing* dapat meningkatkan efisiensi keseluruhan infrastruktur dengan cara mengurangi latensi, mengurangi beban *backhaul*, mendukung layanan mobilitas, dan meningkatkan ketahanan layanan (Mansouri & Babar, 2021).

Paradigma *edge computing* terdiri atas perhubungan antara perangkat-perangkat yang terbatas sumber daya (*smartphone*, *wearable*, SBC), perangkat jaringan (switch, router), dan perangkat-perangkat dengan sumber daya moderat (*base station*), LAN, Cloudlets, dll. Perangkat-perangkat yang terhubung menghasilkan dan mengumpulkan sejumlah besar data yang awalnya diproses dan dianalisis di tepi jaringan. Data-data ini kemudian dapat ditransfer ke *cloud* untuk ekstraksi menggunakan berbagai pendekatan *Machine Learning* dan Optimasi (konveks atau Bayesian). Dengan demikian, *edge computing* melengkapi komputasi *cloud* (Mansouri & Babar, 2021). Gambar 1 mengilustrasikan penjelasan tersebut.

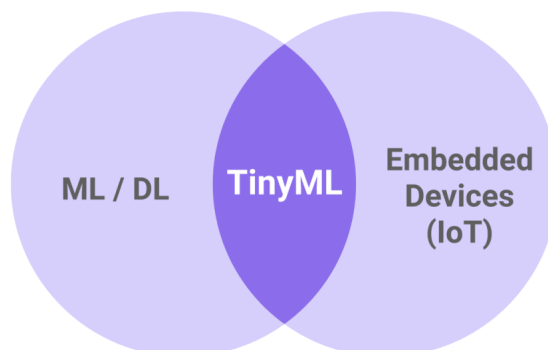
## Cloud Computing vs Edge Computing



Gambar 1 Ilustrasi perbedaan *cloud computing* dengan *edge computing* (*Edge Computing vs. Cloud Computing: What's the Difference?*, 2021)

Seperti halnya *cloud computing*, teknologi fundamental dari paradigma *edge computing* adalah virtualisasi sumber daya yang memisahkan sumber daya perangkat keras dari perangkat lunak untuk menjalankan proses-proses pada beberapa pengguna menggunakan perangkat keras yang sama. Namun, *edge computing* memiliki perbedaan dengan komputasi *cloud* dalam hal lokalitas, layanan berbasis mobilitas, perangkat yang terbatas sumber daya dan heterogen, serta distribusi perangkat yang luas. (Mansouri & Babar, 2021).

### 2.4 TinyML



Gambar 2 Diagram venn korelasi ML, IoT, dan TinyML (Tomas, 2021).

TinyML merupakan pendekatan yang dikembangkan untuk menghubungkan ML dengan perangkat *edge*. TinyML memungkinkan pengimplementasian model DL berukuran kecil ke dalam perangkat *edge* yang memiliki keterbatasan sumber daya yang signifikan, seperti kemampuan komputasi terbatas, kapasitas penyimpanan rendah, atau perangkat berdaya rendah. Tujuan utama TinyML adalah untuk meningkatkan efektivitas sistem *Deep Learning* melalui pengurangan kebutuhan komputasi dan data untuk memfasilitasi kebutuhan *edge* AI dan IoT sebagaimana diilustrasikan pada Gambar 2.

*Deep Neural Network* (DNN) atau *Deep Learning* (DL) yang merupakan bagian dari *Machine Learning* (ML) yang data kerap kali dimanfaatkan untuk mengolah data pada perangkat-perangkat *edge* sering dihadapkan dengan tantangan berupa terbatasnya kapabilitas perangkat-perangkat *edge* bila dibandingkan dengan besarnya model DL, dan tantangan lamanya waktu yang dibutuhkan bagi model untuk membuat keputusan. TinyML memungkinkan *deployment* model DL berukuran kecil ke dalam perangkat *edge* yang memiliki keterbatasan sumber daya yang signifikan (Alajlan & Ibrahim, 2022).

Dengan TinyML, analisis dan interpretasi data dapat dilakukan secara lokal pada perangkat sehingga pengambilan keputusan atau tindakan dapat dilakukan secara *real time*. Model DL yang telah dilatih sebelumnya dapat di-*deploy* ke dalam perangkat *edge* setelah melakukan kompresi model DL dan mengoptimalkan proses inferensi. Misalnya, menggunakan teknik kuantisasi yang merupakan teknik konversi yang mengubah angka *float-point* menjadi angka presisi minimal untuk mengurangi ukuran model DL dengan degradasi akurasi yang minimal. Teknik *pruning* memungkinkan penghapusan struktur dan parameter jaringan yang berlebihan. TinyML memungkinkan berbagai perangkat IoT untuk mengolah data secara lokal tanpa perlu terhubung ke *cloud* untuk memproses data. TinyML menawarkan berbagai keuntungan, termasuk penghematan biaya, pengurangan konsumsi energi, dan peningkatan privasi (Alajlan & Ibrahim, 2022).

## 2.5 Edge Impulse

Edge Impulse merupakan platform operasi pembelajaran mesin (MLOps) berbasis *cloud* yang dikembangkan untuk sistem pembelajaran mesin terintegrasi

dan *edge* ML (TinyML), yang dapat diterapkan pada berbagai target perangkat keras. Edge Impulse diperkenalkan sebagai platform MLOps praktis untuk pengembangan sistem TinyML dalam skala besar dan menyederhanakan siklus desain TinyML dengan mendukung berbagai optimasi perangkat lunak dan perangkat keras untuk menciptakan tumpukan perangkat lunak yang ekstensif dan portabel untuk beragam sistem terintegrasi (Hymel et al., 2022).

Edge Impulse mengintegrasikan ekstraksi fitur *preprocessing* dengan DL. Selain itu, Edge Impulse menciptakan *Edge Optimized Neural* (EON) *Tuner* yang secara otomatis mengeksplorasi ruang pencarian yang ditentukan pengguna, mencakup *preprocessor* dan model ML. Selain itu, dengan kompiler EON, Edge Impulse menyediakan *library* inferensi yang ekstensibel dan portabel, yang dapat diterapkan di berbagai sistem *edge* dan terintegrasi, mengoptimalkan penggunaan sumber daya dengan mengurangi ruang RAM dan flash yang biasanya diperlukan oleh interpreter TensorFlow Lite Micro (TFLM). Compiler ini tidak hanya mengompilasi *neural network* TFLM menjadi *source code* C++ tetapi juga mengeliminasi kebutuhan akan interpreter TFLM dengan menghasilkan kode yang langsung memanggil kernel yang mendasarinya sehingga memungkinkan penghapusan instruksi yang tidak digunakan dan secara signifikan menurunkan penggunaan RAM dan ROM. Edge Impulse mendukung optimasi spesifik perangkat untuk kebutuhan perangkat keras tertentu dengan ekstensibilitas platform memungkinkan penambahan blok pemrosesan, pembelajaran, dan *deployment* kustom untuk mengakomodasi target dan optimasi tambahan sehingga menyesuaikan solusi untuk kemampuan perangkat keras spesifik dan meningkatkan efisiensi keseluruhan implementasi *neural network* pada perangkat *edge* (Hymel et al., 2022).

## 2.6 Bluetooth Low Energy

Bluetooth Low Energy (BLE) merupakan protokol komunikasi radio jarak dekat yang pertama kali digunakan dalam Bluetooth 4.0. BLE dan memiliki kemampuan untuk meminimalkan konsumsi daya dan biaya perangkat pada perangkat-perangkat berdaya rendah bila dibandingkan dengan teknologi Bluetooth sebelumnya. Fitur paling signifikan adalah konsumsi daya yang lebih rendah karena

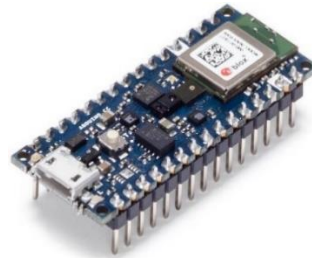
BLE berada dalam mode tidur pada sebagian besar waktu dan hanya membangunkan perangkat ketika aktivitas terjadi. BLE dirancang dengan keadaan *deep sleep* (Duty-Cycle) untuk menggantikan waktu *idle* dari Bluetooth classic. Selain itu, modifikasi BLE terhadap Bluetooth tradisional terletak pada mode *broadcast* dan mode *scanning*. Dalam mode *broadcast*, BLE hanya menggunakan 3 pita frekuensi, sedangkan Bluetooth classic menggunakan 16 hingga 32 pita frekuensi. Perubahan tersebut sangat mengurangi konsumsi daya. (Liu et al., 2021).

Tidak seperti komunikasi Bluetooth classic yang pada dasarnya berbasis koneksi serial asinkron (UART), perangkat-perangkat yang terhubung dengan BLE memiliki salah satu dari dua macam peran, yaitu sebagai perangkat *peripheral* dan perangkat *central*. Perangkat *peripheral* adalah perangkat yang menyalurkan informasi dengan cara mengiklankan (*advertising*) data tersebut kepada radio BLE di sekitarnya. Informasi tersebut akan diakses oleh perangkat *central* bila data yang disajikan merupakan informasi yang dicari. Komunikasi antara perangkat *peripheral* dan *central* melibatkan struktur yang terdiri dari *services* dan *characteristics*, serta menggunakan mekanisme seperti *notify*, *indicate*, *read*, dan *write*. Perangkat *peripheral* bertugas mengumpulkan data dan mengirimkannya ke perangkat *central*. Data ini diatur dalam *services*, yang merupakan kumpulan dari *characteristic*. Setiap *characteristic* memiliki UUID berisi nilai data serta properti yang menentukan bagaimana nilai tersebut dapat diakses, termasuk apakah dapat dibaca (*readable*) atau ditulis (*writable*) (ArduinoBLE, n.d.).

Mekanisme *notify* memungkinkan *peripheral* untuk mengirim pembaruan nilai *characteristic* secara otomatis ke perangkat *central* ketika terjadi perubahan data, tanpa memerlukan perangkat *central* untuk mengirim permintaan *read*. Di sisi lain, *indicate* berfungsi serupa dengan *notify*, tetapi dengan tambahan pengakuan (*acknowledgement*) dari perangkat *central* setelah menerima pembaruan untuk menjamin bahwa data telah diterima dengan sukses. Untuk mekanisme *read*, perangkat *central* mengirim permintaan ke *peripheral* untuk membaca nilai *characteristic* saat itu juga. Mekanisme ini sering digunakan untuk data yang cenderung statis. Sedangkan *write* memungkinkan perangkat *central* untuk mengirim data ke *peripheral* guna mengubah nilai dari *characteristic*, mekanisme

ini sering kali digunakan untuk mengontrol perangkat atau mengubah pengaturannya. (*ArduinoBLE*, n.d.).

## 2.7 Arduino Nano 33 BLE Sense



Gambar 3 Arduino Nano 33 BLE Sense (Machine Learning on Arduino Nano 33 BLE Sense, 2021)

Arduino Nano 33 BLE Sense dibangun menggunakan pondasi mikrokontroler nRF52840 dan mengoperasikan sistem operasi Arm Mbed. Arduino Nano 33 BLE Sense memiliki kapasitas penyimpanan sebesar 1 MB dan RAM 256 KB. Modul NINA B306 yang tersemat memungkinkan pengguna untuk dapat saling terhubung menggunakan teknologi Bluetooth Low Energy sejauh kurang lebih 50 meter dengan kapasitas transmisi 2 Mbps. Selain itu, Arduino Nano 33 BLE Sense juga dilengkapi dengan berbagai macam sensor untuk mendeteksi warna, kedekatan dan gestur (APDS9960), gerakan (LSM9DS1), suhu dan kelembaban (HTS221), audio (MP34DT05), dan tekanan barometrik (LPS22HB). Papan ini beroperasi pada daya 3.3 V dan 5 V yang optimal untuk penggunaan berdaya rendah jangka panjang (*Arduino Nano 33 BLE Sense*, n.d.).

## 2.8 Signal Vector Magnitude / Sum Vector Magnitude (SMV)

Akselerometer 3 sumbu adalah sebuah sensor yang menghasilkan estimasi percepatan bernilai nyata di sepanjang sumbu x, y, dan z. Sensor ini mengukur percepatan dan menghasilkan proyeksi vektor percepatan yang direpresentasikan dalam sistem koordinat 3D (Gjoreski & Gams, 2011). Akselerasi atau percepatan adalah sebuah istilah yang diberikan pada suatu proses dimana terdapat perubahan kecepatan (*velocity*). Kecepatan merupakan kelajuan (*speed*) dan arah (*direction*), sehingga terdapat tiga cara untuk berakselerasi, yaitu dengan mengubah kelajuan, mengubah arah, atau mengubah keduanya. Secara spesifik, percepatan



didefinisikan sebagai derajat perubahan pada kecepatan, persamaan 1 menjelaskan hal ini.

$$a = \frac{dv}{dt} = \frac{v_b - v_a}{dt} \quad (1)$$

Dimana akselerasi ( $a$ ) sama dengan selisih antara kecepatan awal ( $v_a$ ) dan kecepatan akhir ( $v_b$ ) dibagi dengan waktu ( $dt$ ) yang dibutuhkan untuk mengubah kecepatan dari a ke b (*What Is Acceleration?*, 2015).

Satuan percepatan adalah meter per detik kuadrat ( $m/s^2$ ). Namun, sensor akselerometer biasanya menyatakan pengukuran dalam “g” atau gravitasi. Satu “g” (1g) sama dengan kurang lebih  $9.8 m/s^2$ . Karena gravitasi bumi, semua benda mengalami tarikan gravitasi ke arah pusat bumi. Ketika akselerometer dalam keadaan diam, satu-satunya gaya yang mempengaruhi sensor adalah gravitasi bumi. Akibatnya, semua benda mengalami percepatan 1 g (Gjoreski & Gams, 2011). Menurut Su dan Twu dalam penelitian mereka, penggunaan akselerometer untuk sistem deteksi berbasis akselerometer disarankan untuk diletakkan di saku pinggang alih-alih di tangan atau kaki karena pusat gravitasi manusia berada sekitar tulang panggul (Su & Twu, 2020). Akselerasi tiga dimensi direpresentasikan oleh persamaan 2:

$$\vec{A} = \frac{d\vec{v}}{dt} = (\vec{g} + \vec{l}), \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} = \begin{pmatrix} g_x + l_x \\ g_y + l_y \\ g_z + l_z \end{pmatrix} \quad (2)$$

Dimana akselerasi ( $\vec{A}$ ), akselerasi akibat gravitasi ( $\vec{g}$ ), dan akselerasi linier resultan ( $\vec{l}$ ) diukur dalam  $m/s^2$  (Janidarmian et al., 2017).

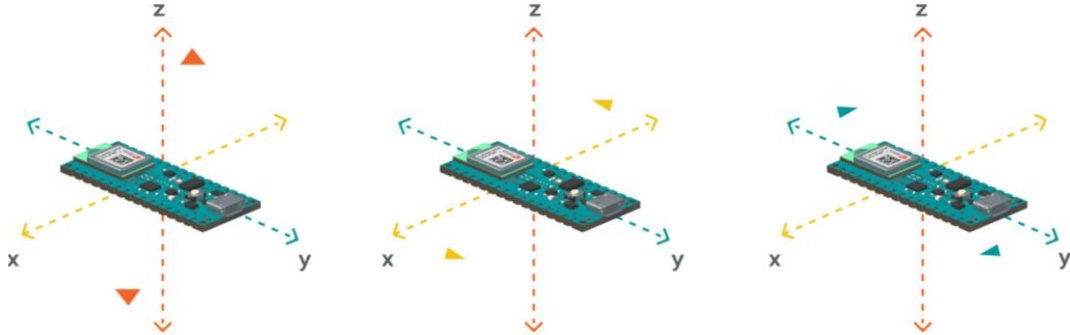
Magnitudo merujuk pada ukuran atau panjang dari sebuah vektor terlepas dari arah vektor tersebut. Rumus magnitudo dijabarkan pada persamaan 3 di bawah:

$$\vec{m} = \sqrt{x^2 + y^2} \quad (3)$$

Dimana x dan y merupakan komponen-komponen yang merepresentasikan jarak atau perpindahan dari vektor tersebut pada masing-masing sumbu dari asalnya ke titik akhirnya. Perpindahan x dari asalnya ke titik akhirnya akan terjadi sepanjang sumbu x saja, begitu pula dengan sumbu y, baik ke arah positif maupun ke arah negatif. Bila terdapat tiga sumbu, maka persamaan magnitudo dijabarkan sebagai berikut:

$$\vec{m} = \sqrt{x^2 + y^2 + z^2} \quad (4)$$

Dimana x, y, dan z merepresentasikan tiap sumbu tiga dimensi sebagaimana yang ditampilkan pada Gambar 4.



Gambar 4 Ilustrasi akselerasi tiga dimensi (Alushi, 2022)

*Signal Vector Magnitude*, yang seringkali disingkat sebagai SMV untuk menghindari kekeliruan dengan *Support Vector Machine* (SVM), adalah fitur domain waktu yang menghitung total intensitas percepatan (magnitudo) yang dihasilkan oleh data percepatan triaksial. Hasilnya dapat menunjukkan seberapa signifikan perubahan percepatan dalam suatu momen waktu terlepas dari arahnya atau pada sumbu mana perubahan tersebut terjadi. SMV merupakan pengaplikasian spesifik magnitudo dalam konteks pemrosesan sinyal, yaitu dengan cara menggabungkan sinyal-sinyal dari ketiga sumbu akselerometer untuk menghasilkan sebuah nilai skalar yang merepresentasikan derajat intensitas suatu pergerakan pada suatu waktu. Rumus SMV dijabarkan pada persamaan x di bawah:

$$SMV[i] = \sqrt{|A_{xi}|^2 + |A_{yi}|^2 + |A_{zi}|^2} \quad (5)$$

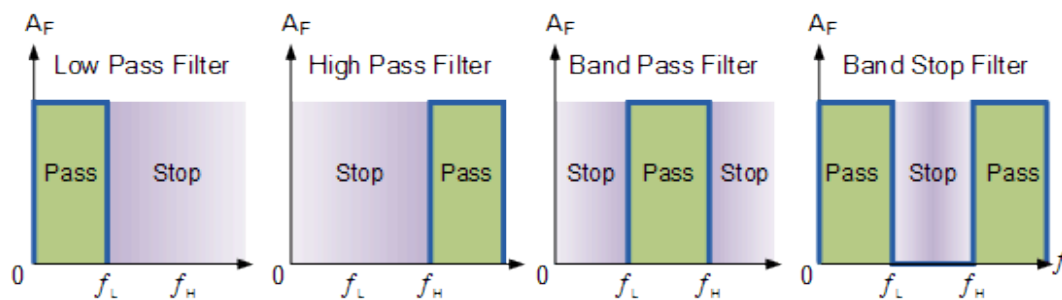
Dimana  $SMV[i]$  adalah magnitudo vektor akselerasi ke-i sedangkan  $A_{xi}$ ,  $A_{yi}$ , dan  $A_{zi}$  mendefinisikan komponen-komponen dari vektor akselerasi untuk sampel ke-i pada masing-masing vektor x, y, dan z (Al-Kababji et al., 2021; Casilari et al., 2020).

## 2.9 Filter

Filter digunakan untuk menyaring masukan menurut rentang frekuensi sinyal yang dibolehkan (*pass*) sembari memblokir (*stop*) sisanya. Desain filter yang paling umum digunakan adalah:

1. Filter *Low Pass*: filter *low pass* hanya membolehkan sinyal frekuensi rendah (dari 0 Hz hingga frekuensi *cut-off*) untuk melewatinya sambil memblokir sinyal yang lebih tinggi.
2. Filter *High Pass*: filter *high pass* hanya membolehkan sinyal frekuensi tinggi (dari frekuensi *cut-off* dan selebihnya) untuk melewatinya sambil memblokir sinyal yang lebih rendah.
3. Filter *Band Pass*: filter *band pass* membolehkan sinyal yang berada dalam pengaturan *band* frekuensi tertentu sambil memblokir frekuensi yang lebih rendah dan lebih tinggi daripada kedua sisi *band* frekuensi ini.
4. Filter *Band Stop*: filter ini membolehkan sinyal yang lebih rendah dan lebih tinggi daripada rentang *band* frekuensi tertentu sambil memblokir sinyal pada rentang tengah yang ditentukan.

Gambar 5 mengilustrasikan jenis-jenis filter yang telah disebutkan.



Gambar 5 Jenis-jenis filter (Storr, 2013)

Gjoreski & Gams dalam penelitian mereka menggunakan filter low-pass guna memperhalus data akselerasi untuk pengenalan aktivitas agar bagian data akselerasi yang disebabkan oleh gravitasi tersaring dari data pergerakan akselerometer itu sendiri (Gjoreski & Gams, 2011). Rumus filter *low-pass* yang digunakan oleh Gjoreski dan Gams adalah sebagai berikut.

$$\alpha = 0,8$$

$$Low\_X = \alpha * Previous\_X + (1 - \alpha) * Current\_X \quad (6)$$

Dimana *Low\_X* merupakan nilai yang telah melewati filter, *Previous\_X* merupakan nilai *x* sebelumnya (*x-1*), dan *Current\_X* merupakan nilai *x* saat ini (Gjoreski & Gams, 2011).

## 2.10 UniMiB SHAR

UniMiB SHAR merupakan sebuah dataset sampel akselerasi yang diperoleh menggunakan *smartphone* Android yang dirancang untuk *human activity recognition (HAR)* atau pengenalan aktivitas manusia, dan deteksi jatuh. Dataset ini mencakup 11.771 sampel dari *Activities of Daily Living (ADL)* dan *Fall* (jatuh). Sampel dibagi dalam 17 kelas terperinci yang dikelompokkan dalam dua kelas kasar: 9 jenis aktivitas sehari-hari (ADL) dan 8 jenis jatuh. Perangkat yang digunakan untuk akuisisi data merupakan Samsung Galaxy Nexus I9250 yang memiliki akselerometer Bosch BMA220, merekam data akselerasi dengan frekuensi sampel 50 Hz untuk setiap aktivitas. Tiap vektor data akselerometer terdiri atas 3 vektor dengan 151 nilai (vektor berukuran  $1 \times 453$ ), satu untuk setiap arah percepatan. Dataset ini terdiri dari 11,771 sampel yang mendeskripsikan ADL (7759) dan jatuh (4192).

Terdapat 30 subjek yang melakukan simulasi ADL dan jatuh yang dengan rentang usia dari 18 hingga 60 tahun dan tinggi badan 160 cm hingga 190 cm. Protokol akuisisi data yang dilakukan yaitu dengan mengenakan *smartphone* yang akan merekam akselerasi dan suara di kantong celana subjek. Subjek menepuk tangan sebelum dan sesudah melakukan tiap aktivitas yang ditentukan. Rekaman audio membantu mengidentifikasi waktu mulai dan berhenti untuk tiap aktivitas. Window sinyal sepanjang 3 detik diekstraksi dari tiap data berpusat pada sebuah lonjakan dimana magnitudo sinyal  $mt$  pada waktu  $t > 1.5$  g dan  $mt-1$  pada waktu  $t-1 < 1.5$  g. Segmentasi tersebut digunakan alih-alih menggunakan *overlapped sliding windows* karena dataset ini berfokus pada pengenalan kegiatan dan jatuh berbasis gerakan. Windows berdurasi 3 detik dipilih karena rerata manusia berjalan dengan 90 hingga 130 langkah per menit dan setidaknya satu siklus jalan (dua langkah) terjadi (Micucci et al., 2017).

## 2.11 Deep Neural Network (DNN)

Jaringan Saraf Tiruan atau *Artificial Neural Network (ANN)* adalah mesin yang dirancang untuk melakukan tugas-tugas tertentu dengan meniru cara kerja otak manusia, dan membangun jaringan saraf yang terdiri dari ratusan atau bahkan ribuan neuron buatan atau unit pemrosesan. Implementasi praktis jaringan saraf

dimungkinkan karena *neural network* merupakan sistem komputasi paralel masif yang terdiri dari sejumlah besar unit pemrosesan dasar (neuron) dan bobot sinapsis. Tugas algoritma pembelajaran tersebut terdiri dari pengubahan bobot sinapsis jaringan secara berurutan untuk mencapai suatu tujuan tertentu (Montesinos-López et al., 2022).

Montesinos-López et al. mendefinisikan *deep learning* sebagai generalisasi dari ANN di mana lebih dari satu *hidden layer* digunakan yang menyiratkan bahwa lebih banyak neuron digunakan untuk mengimplementasikan model. Oleh karena itu, ANN dengan banyak *hidden layer* disebut dengan *Deep Neural Network* (DNN) dan praktik pelatihan jenis jaringan ini disebut *deep learning* (DL), yang merupakan cabang dari *machine learning* statistik di mana topologi *multilayer* digunakan untuk memetakan hubungan antara variabel input (variabel independen) dan variabel respons (hasil). (Montesinos-López et al., 2022).

### 2.11.1 Batch Size

*Batch size* merupakan sebuah *hyperparameter* yang mendefinisikan jumlah sampel yang akan diproses sebelum pembaruan parameter model internal. Umpamakan sebuah *batch* sebagai *for-loop* yang mengulang satu atau lebih sampel dan membuat prediksi. Setelah itu, prediksi yang telah dihasilkan kemudian dibandingkan dengan variabel keluaran yang diharapkan, lalu sebuah *error* dihitung. Dari *error* ini, algoritma pembaruan digunakan untuk meningkatkan model. Sebuah set data pelatihan dapat dibagi menjadi satu atau lebih batch (Brownlee, 2022).

### 2.11.2 Epoch

*Epoch* merupakan sebuah *hyperparameter* yang mendefinisikan jumlah perulangan yang akan dilalui oleh algoritma pembelajaran untuk memproses seluruh dataset pelatihan. Satu *epoch* berarti bahwa setiap sampel dalam dataset pelatihan memiliki satu kali kesempatan untuk memperbarui parameter model internal. Sebuah *epoch* terdiri dari satu atau beberapa batch (Brownlee, 2022).

Umpamakan sebuah *for-loop* pada jumlah *epoch* di mana setiap perulangan dilakukan pada kumpulan data pelatihan. Di dalam *for-loop* ini terdapat *nested for-loop* lainnya yang mengulang setiap *batch* sampel, di mana satu *batch* memiliki

jumlah sampel “batch size” yang ditentukan. Jumlah *epoch* biasanya besar, sering kali berjumlah ratusan atau ribuan agar algoritma pembelajaran berjalan hingga kesalahan dari model dapat diminimalkan (Brownlee, 2022).

### 2.11.3 Learning Rate

*Learning Rate* merupakan sebuah *hyperparameter* yang mengatur seberapa banyak kita menyesuaikan *weight* dari *network* dengan mengacu pada *loss gradient*. Semakin rendah nilainya, semakin lambat pergerakan di sepanjang *slope* penurunan. Meskipun menggunakan *learning rate* yang rendah memungkinkan untuk tidak melewati *local minima*, waktu yang dibutuhkan untuk konvergensi akan lebih lama, terutama jika terjebak di daerah *plateau* (Zulkifli, 2018).

*Learning rate* mengontrol laju atau kecepatan model belajar. Secara spesifik, ia mengontrol jumlah *error* yang diproporsikan dengan bobot model yang diperbarui setiap kali bobot tersebut diperbarui. Dengan konfigurasi *learning rate* yang sempurna, model akan belajar untuk membuat prakiraan terbaik dari fungsi dengan sumber daya yang tersedia (jumlah *layer* dan jumlah node per *layer*) dalam jumlah *epoch* tertentu. Secara umum, *learning rate* yang besar memungkinkan model untuk belajar lebih cepat meski set bobot akhir akan kurang optimal. *Learning rate* yang kecil memungkinkan model untuk mempelajari set bobot yang lebih optimal atau bahkan optimal secara global walau membutuhkan waktu pelatihan yang lebih lama (Brownlee, 2019).

### 2.11.4 Confusion Matrix

*Confusion Matrix* adalah tabel silang (*cross table*) yang menunjukkan jumlah kemunculan antara dua nilai kelas yaitu klasifikasi yang sebenarnya atau bernilai aktual dan klasifikasi prediksi, sebagaimana yang ditunjukkan pada Tabel 1, kolom-kolom pada tabel tersebut menunjukkan prediksi model sedangkan baris-baris menunjukkan klasifikasi yang sebenarnya (Grandini et al., 2020).

Tabel 1 *Confusion matrix*

		Kelas Prediksi	
		Positif	Negatif
Kelas Aktual	Positif	TP	FN
	Negatif	FP	TN

Dimana:

TP (*True Positive*): nilai aktual positif dan model memprediksi positif.

TN (*True Negative*): nilai aktual negatif dan model memprediksi negatif.

FP (*False Positive*): nilai aktual negatif, namun model memprediksi positif.

FN (*False Negative*): nilai aktual positif, namun model memprediksi negatif.

*Confusion matrix* membantu dalam memahami sejauh mana model dapat membedakan antara kelas positif dan negatif serta seberapa sering model menghasilkan prediksi yang salah. Informasi dari *confusion matrix* juga dapat digunakan untuk menghitung metrik evaluasi seperti akurasi, presisi, *recall*, dan *F1-score*. Dalam penelitian ini, digunakan perhitungan akurasi. Akurasi adalah perhitungan jumlah prediksi benar yang dihasilkan oleh model untuk seluruh dataset uji. Akurasi dapat diperoleh dengan menggunakan persamaan x di bawah:

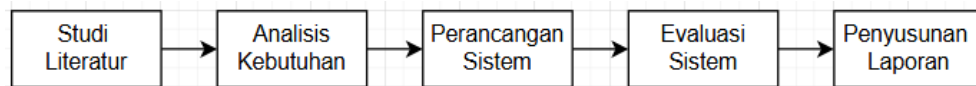
$$Akurasi = \frac{TP + TN}{TP + FP + TN + FN} \times 100\% \quad (7)$$

Dimana TP dan TN pada pembilang adalah elemen-elemen yang diklasifikasikan dengan benar oleh model dan elemen-elemen tersebut berada pada diagonal utama *confusion matrix*, sementara penyebut juga memperhitungkan semua elemen di luar diagonal utama yang telah diklasifikasikan secara keliru oleh model (Grandini et al., 2020).

## BAB 3 METODE PENELITIAN/PERANCANGAN

### 3.1 Tahapan Penelitian

Tahapan penelitian ini dijelaskan sebagai berikut:



Gambar 6 Diagram tahapan penelitian

1. Studi literatur, pada tahap ini dilakukan pengkajian literatur-literatur yang berhubungan dengan deteksi jatuh, implementasi BLE, Edge Impulse, dan *edge computing* beserta dokumentasi penelitian-penelitian terkait.
2. Analisis kebutuhan sistem, pada tahap ini dilakukan analisis untuk menentukan berbagai alat dan metode untuk menunjang keberhasilan sistem.
3. Perancangan sistem, rancangan sistem dibuat dan sistem dibangun sesuai rancangan tersebut.
4. Evaluasi sistem, sistem yang telah dibangun lalu diuji dan dievaluasi.
5. Penyusunan laporan, laporan disusun berdasarkan hasil penelitian.

### 3.2 Waktu dan Lokasi Penelitian

Penelitian ini dilaksanakan di Laboratorium PCIOT Departemen Teknik Informatika, Fakultas Teknik Universitas Hasanuddin, Kabupaten Gowa sejak penelitian ini diproposalkan hingga pelaporan hasil penelitian.

### 3.3 Instrumen Penelitian

#### 3.3.1 Hardware

1. Laptop (AMD FX-9830P, AMD RX 460)
2. Arduino Nano 33 BLE Sense \* 2
3. Modul Buzzer
4. Baterai LiPo 18650 3,7 volt
5. Modul Step Up / Power Booster MT3608
6. Kabel *jumper*



### 3.3.2 Software

1. Windows 10 x64
2. MinGW-W64
4. Arduino IDE 2.2.1
5. IDLE Python 3.11.1 / Command Prompt
6. Microsoft Edge 64-bit

### 3.3.3 Library

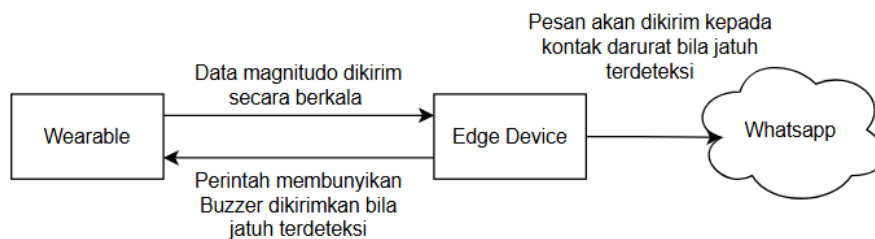
1. Numpy 1.26.0
2. Pandas 2.1.1
3. Pywhatkit 5.4
4. Pyserial 3.5
5. Pyautogui
6. ArduinoBLE 1.3.6
7. Arduino\_LSM9DS1

### 3.3.4 Dataset

1. UniMIB SHAR

## 3.4 Rancangan Sistem

Sistem pada penelitian ini meliputi sebuah *wearable*, perangkat *edge*, dan *cloud*. Gambar 7 di bawah menunjukkan skema kerja sistem secara garis besar.

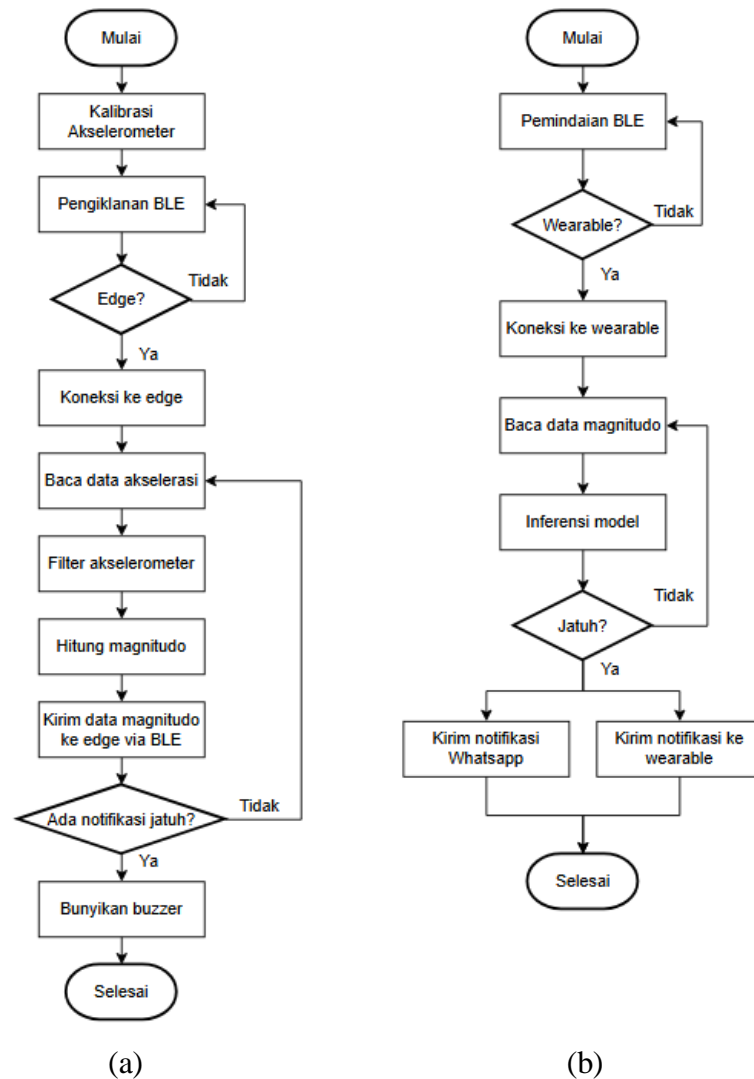


Gambar 7 Diagram blok sistem

Perangkat *wearable* sebagai perangkat *peripheral* memiliki komponen utama sebuah Arduino Nano 33 BLE Sense dengan IMU tersemat untuk merekam sinyal akselerometer dan menghitung nilai magnitudo sembari berkomunikasi dengan perangkat *edge*. Sedangkan perangkat *edge* sebagai perangkat *central*

terdiri dari sebuah mikrokontroller yang terhubung ke laptop melalui USB untuk melakukan pemrosesan dan mengirimkan pesan Whatsapp.

Selanjutnya pada keseluruhan penelitian ini, perangkat *wearable* disebut sebagai “wearable” dan perangkat *edge* disebut sebagai “edge”.

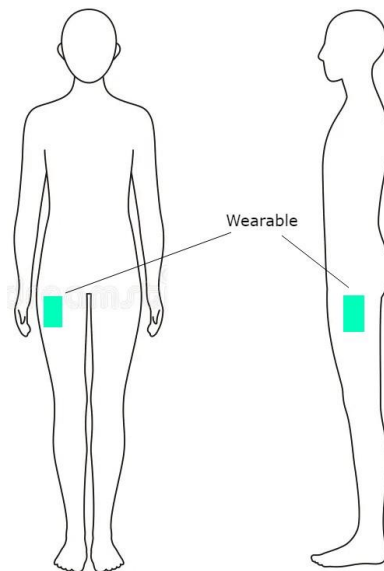


Gambar 8 Alur kerja (a) perangkat edge dan (b) perangkat wearable

Gambar 8 menunjukkan skema kerja sistem secara lebih terperinci. Langkah pertama yang dilakukan wearable setelah setup yaitu mengkalibrasi akselerometer. Wearable mengiklankan *local name* dan edge memindai perangkat BLE di sekitarnya. Bila edge mendapati frekuensi BLE dengan *local name* wearable yang telah ditentukan, edge akan menginisiasi perhubungan dengan perangkat tersebut lalu wearable akan menerima permintaan perhubungan. Setelah kedua perangkat saling terkoneksi, wearable mulai membaca data akselerasi tiga sumbu dari IMU

dan menghitung magnitudo dari nilai ketiga sumbu tersebut lalu mengirimkan data tersebut ke edge melalui BLE. Model yang ada dalam edge akan membuat keputusan berdasarkan data yang diterima, yaitu keputusan ada tidaknya kejadian jatuh. Bila model ML mendeteksi adanya kejadian jatuh, edge akan mengirim pesan Whatsapp kepada kontak darurat dan notifikasi kepada wearable untuk membunyikan buzzer.

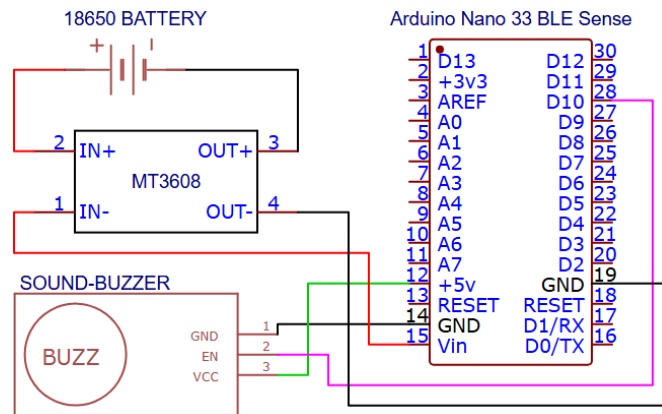
Wearable pada penelitian ini dikenakan dalam kantong celana sebagaimana anjuran Su & Twu agar sekaligus menyesuaikan dengan dataset UniMIB SHAR yang berisi data akselerasi yang diakuisisi dengan akselerometer dalam kantong celana pemakai seperti yang diilustrasikan pada gambar 9 di bawah.



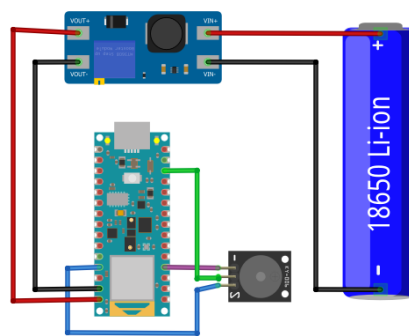
Gambar 9 Letak pemakaian *wearable*

### 3.4.1 Rancangan Alat

Gambar 10 menunjukkan komponen-komponen pada perangkat wearable, yaitu: baterai, modul booster, Arduino Nano 33 BLE Sense, dan buzzer. Arduino Nano 33 BLE Sense dapat dinyalakan melalui port USB micro-b dan pin VIN dengan tegangan 4,5 - 21 volt. Arduino akan diperantarai oleh booster agar tegangan dari baterai yang berjumlah 3,7 volt dapat dinaikkan menjadi 5 volt.



(a)



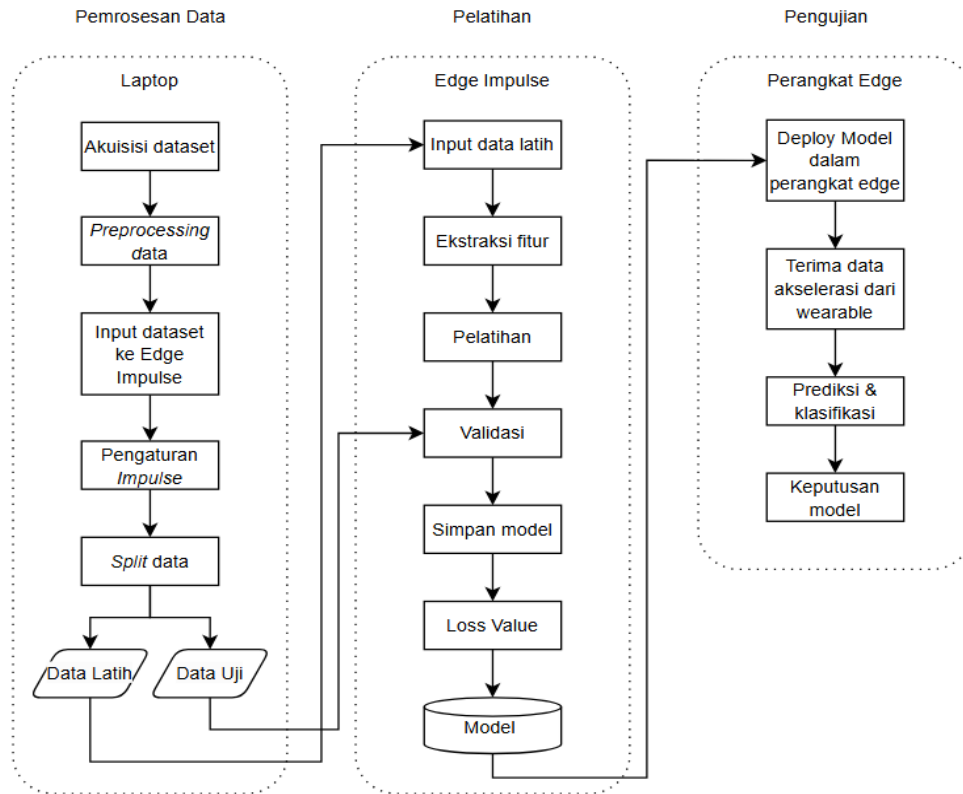
(b)

Gambar 10 (a) EDA wearable dan (b) desain fritzing wearable

Kutub positif baterai dihubungkan ke pin IN+ booster, sedangkan kutub negatif dihubungkan ke pin IN-. Tegangan keluaran booster diatur dengan cara menghubungkan kedua pin OUT ke multimeter untuk mengukur tegangan keluaran sambil memutar potensiometer hingga tegangan keluaran bernilai 5 volt. Selanjutnya pin OUT+ booster dihubungkan ke pin VIN Arduino dan pin OUT-booster dihubungkan ke pin *Ground* Arduino. Modul *buzzer* akan mengambil daya dari Arduino melalui pin VCC yang dihubungkan ke pin 5v Arduino, pin GND *buzzer* dihubungkan ke pin GND Arduino, dan pin EN *buzzer* dihubungkan ke pin D10 arduino.

### 3.4.2 Pembangunan Model

Model pada penelitian ini dibangun di Edge Impulse. Pengguna Edge Impulse melakukan tiga hal untuk membangun model, yaitu mengunggah data, mengatur *Impulse*, dan mengatur *deployment* untuk model yang telah dibuat. Gambar berikut mengilustrasikan skema pembangunan dan pengujian model.

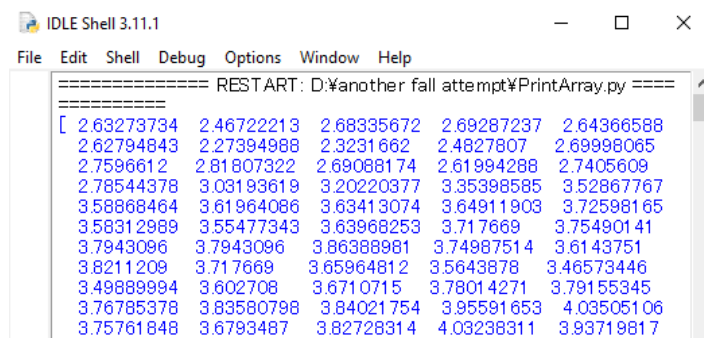


Gambar 11 Alur pembangunan dan pengujian model

Dataset yang digunakan yaitu UniMIB SHAR yang didapatkan dari [kaggle](#). Dataset yang diunduh berupa file .npy kemudian dimodifikasi agar dapat diunggah di Edge Impulse. *Uploader* Edge Impulse hanya menerima format file CSV, CBOR, JSON, WAV, JPG, dan PNG. Pada penelitian ini, dataset akan diunggah ke Edge Impulse dalam bentuk CSV. Dari *flowchart* yang tertera di atas, proses pembangunan model diuraikan sebagai berikut.

### 1. Konversi Dataset

Dataset UniMIB SHAR yang berupa *array* multidimensi dalam tipe file .npy dimodifikasi agar dapat diunggah di Edge Impulse. Pada tahap ini dilakukan proses untuk memisahkan tiga peristiwa berbeda yang ada pada satu sub-*array* menjadi bagian tersendiri. Gambar 12 dan 13 menunjukkan isi salah satu sub-*array* data jatuh dari dataset dan visualisasi sebuah data yang berisi tiga peristiwa jatuh berbeda.



```

IDLE Shell 3.11.1
File Edit Shell Debug Options Window Help
===== RESTART: D:\another fall attempt\PrintArray.py =====
=====
[ 2.63273734  2.46722213  2.68335672  2.69287237  2.64366588
  2.62794843  2.27394988  2.3231662  2.4827807  2.69998065
  2.7596612  2.81807322  2.69088174  2.61994288  2.7405609
  2.78544378  3.03193619  3.20220377  3.35398585  3.52867767
  3.58868464  3.61964086  3.63413074  3.64911903  3.72598165
  3.58312989  3.55477343  3.63968253  3.717669  3.75490141
  3.7943096  3.7943096  3.86388981  3.74987514  3.6143751
  3.8211209  3.717669  3.65964812  3.5643878  3.46573446
  3.49889994  3.602708  3.6710715  3.78014271  3.79155345
  3.76785378  3.83580798  3.84021754  3.95591653  4.03505106
  3.75761848  3.6793487  3.82728314  4.03238311  3.93719817

```

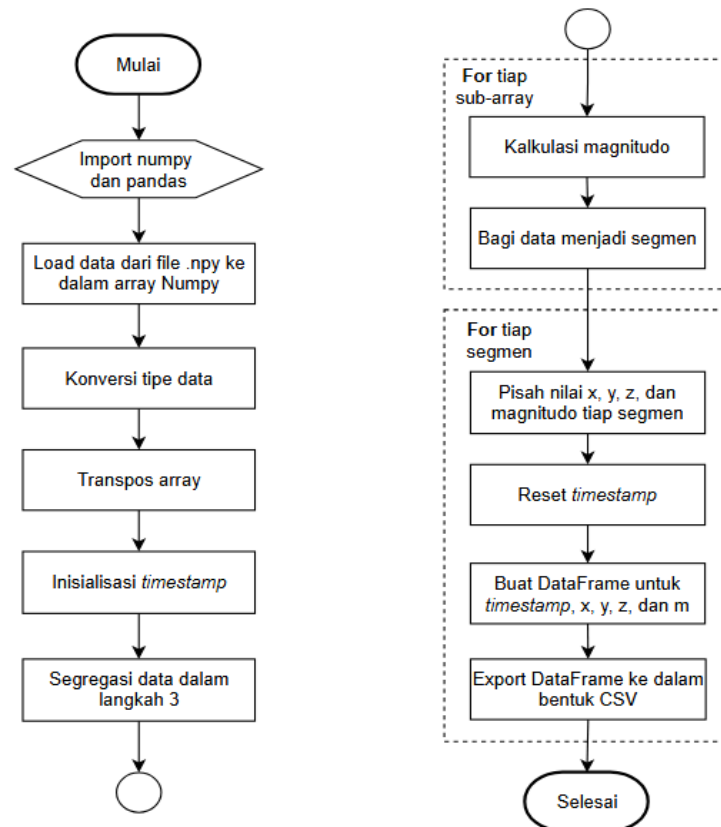
Gambar 12 Isi salah satu sub-array



Gambar 13 Visualisasi data UniMIB SHAR

Gambar 13 menunjukkan sebuah grafik dari data mentah *fall* pada sub-array ke 562. Terdapat 3 macam gelombang yang dapat dilihat, yaitu x (merah), y (hijau), dan z (biru). Dapat dilihat bahwa pada milidetik ke-3000 dan 6000 terdapat lonjakan kecil yang mengisyaratkan transisi dari data jatuh pertama, ke data jatuh kedua, dan data jatuh ketiga. Oleh karena itu, dibuatlah sebuah skema program untuk mengolah dataset yang berupa *array* multidimensi dalam file *.npy* menjadi beberapa file CSV bagi tiap sub-array. Tiap sub-array mengandung data berdurasi 9000 ms, sedangkan durasi tiap data jatuh dan ADL yaitu 3000 ms.

Oleh karena itu, dibuatlah program untuk mengkonversi dataset sesuai kebutuhan yang telah dijelaskan di atas menggunakan python. *Flowchart* pada Gambar 14 menunjukkan alur program tersebut beserta penjelasan lebih lengkap mengenai proses yang telah disebutkan.



Gambar 14 Alur kerja program untuk konversi dataset

Pertama, program memuat data dari file .npz ke dalam *array* NumPy. Data tersebut kemudian dikonversi ke tipe data float64 untuk memastikan presisi numerik, lalu data tersebut ditranspos. Selanjutnya, program menghasilkan rentang angka yang dimulai dari 0 hingga 9060 dengan kenaikan 20 sebagai nilai *timestamp*; nilai 9060 didapatkan dari 453 (panjang data) dikali 20. Kenaikan sebesar 20 secara inkremental dilakukan karena terdapat 50 data per detik, yang berarti setiap data mewakili 20 milidetik untuk mencapai total satu detik.

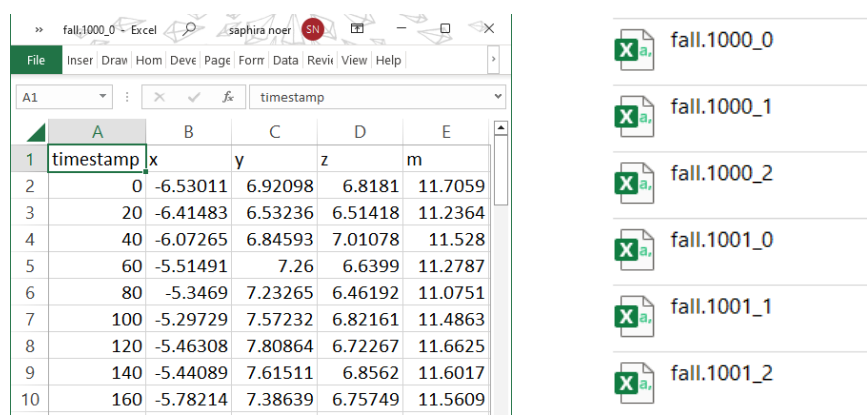
Program kemudian mengiterasi data dalam interval 3 untuk melakukan segregasi nilai-nilai x, y, dan z; data urutan ke-1, ke-4, ke-7, dan seterusnya diambil sebagai nilai x; data urutan ke-2, ke-5, ke-8, dan seterusnya diambil sebagai nilai y; serta data urutan ke-3, ke-6, ke-9, dan seterusnya diambil sebagai nilai z. Untuk setiap set tiga data aksial, dibuat sebuah DataFrame Pandas yang mencakup data yang diekstrak bersama dengan seri *timestamp* yang dihasilkan, dimana seri *timestamp* ini

ditempatkan sebagai kolom pertama. Kolom-kolom tersebut diberi label ‘*timestamp*’, ‘x’, ‘y’, dan ‘z’.

Selanjutnya, terjadi *looping* untuk tiap sub-*array* dimana kolom baru berlabel “m” dibuat berisikan magnitudo dari nilai x, y, dan z di baris yang sama. Setelah itu, program menghitung jumlah baris yang merepresentasikan 3000 milidetik data lalu seluruh data dipisah menjadi 3 bagian, tiap segmen berisi 150 baris data.

Selanjutnya terjadi *looping* untuk tiap segmen. Nilai-nilai x, y, z, dan m pada tiap-tiap segmen akan diisolasi sembari program membuat sebuah *array timestamp* baru untuk tiap segmen dimulai dari angka 0. Dataframe baru akan dibuat dengan kolom ‘*timestamp*’, ‘x’, ‘y’, ‘z’, dan ‘m’. DataFrame akan dieskspor menjadi file CSV dengan format nama <kelas>.<index sub-*array*>\_<index segmen>.csv. Hasil konversi berupa file-file yang mengandung sepertiga dari isi nilai sebuah sub-*array* tertentu. Format penamaan file menunjukkan informasi tersebut, yaitu <kelas>.<index sub-*array*>\_<index segmen>.

Gambar 15 menunjukkan isi file dataset sesudah dikonversi beserta folder berisi data-data satu kelas yang telah dikonversi; pada gambar tersebut terdapat data kelas “fall” pada sub-*array* urutan ke-1000 dan 1001 dengan index 0, 1, dan 2, masing-masing merepresentasikan segmen pertama, segmen kedua, dan segmen ketiga.



(a)

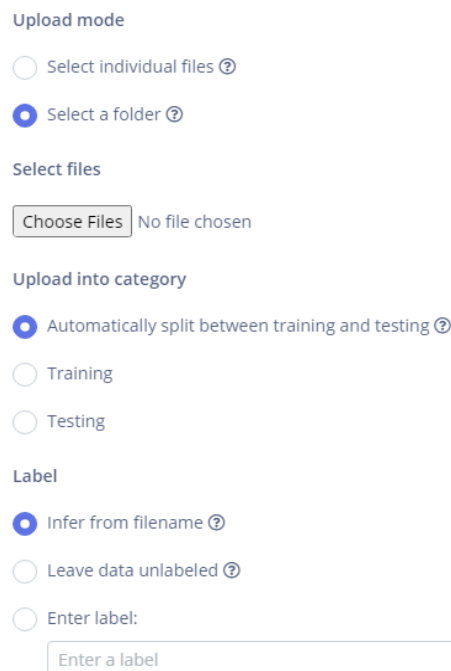
(b)

Gambar 15 Hasil konversi dataset: (a) isi file CSV dan (b) format nama file setelah konversi.



## 2. Pengunggahan dataset ke edge impulse

‘Edge Impulse Studio Uploader’ merupakan fitur pengunggah data di Edge Impulse yang memungkinkan pengguna untuk mengunggah dataset secara massal, yaitu dengan mengunggah isi satu folder. Namun, pengguna juga diberi kebebasan untuk memilih file yang akan diunggah dengan cara menyeleksi file secara individual. Selain itu, pengguna dapat memilih opsi “Automatically split between training and testing” agar dataset yang diunggah secara otomatis dialokasikan untuk pelatihan dan pengujian model. *Uploader* ini juga didukung dengan kapabilitas pelabelan otomatis lewat nama file dengan opsi “Infer from filename”. Format nama file agar dapat dilakukan pelabelan otomatis yaitu <nama label>.<id unik>.csv. Jika suatu file yang diunggah diberi nama “Label01.Sampel02.csv”, maka Edge Impulse akan menetapkan “Label01” sebagai nama label dan “Sampel02” sebagai nama sampel untuk data tersebut. Gambar 16 menunjukkan tampilan *uploader*.



The image shows the 'Edge Impulse Studio Uploader' interface with the following settings:

- Upload mode:**
  - ☐ Select individual files ?
  - ☒ Select a folder ?
- Select files:**
  - Choose Files | No file chosen
- Upload into category:**
  - ☒ Automatically split between training and testing ?
  - ☐ Training
  - ☐ Testing
- Label:**
  - ☒ Infer from filename ?
  - ☐ Leave data unlabeled ?
  - ☐ Enter label:
 

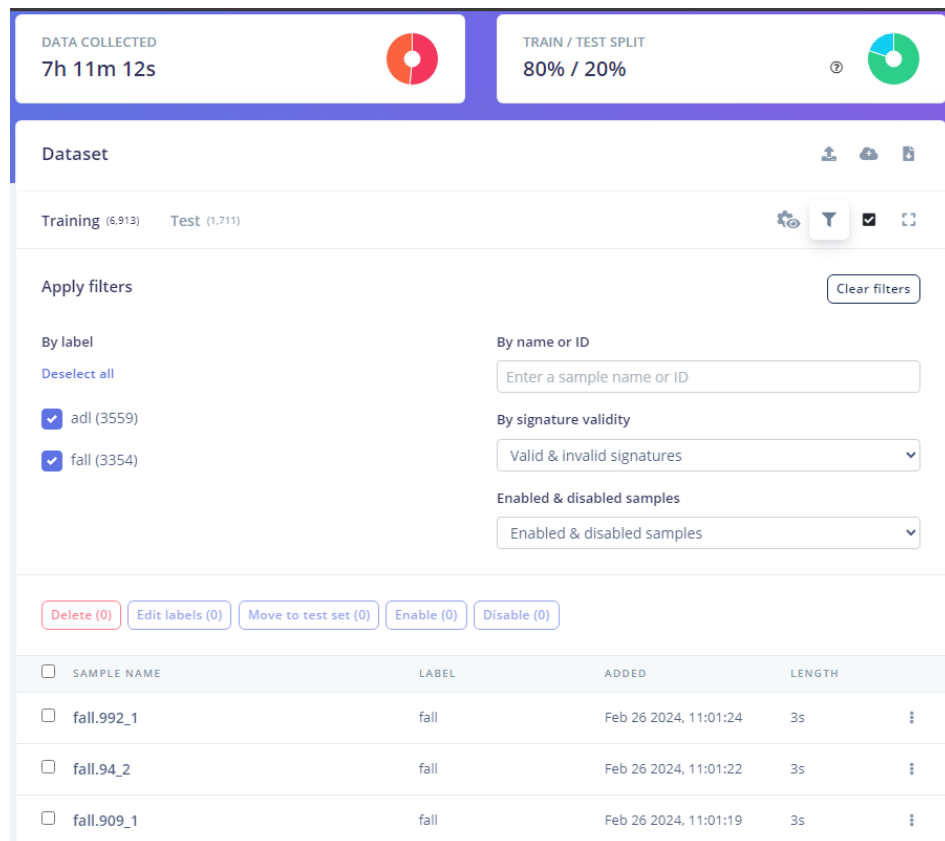
Enter a label

Gambar 16 Opsi pada ‘Edge Impulse Studio Uploader’

Pada penelitian ini, format nama data CSV yang dikonversi, yaitu <kelas>.<index sub-array>\_<index segmen>.csv mengikuti panduan penamaan dari edge impulse, dimana <kelas> dikategorikan sebagai nama

label dan  $\langle \text{index sub-array} \rangle_{\langle \text{index segmen} \rangle}$  dikategorikan sebagai nama sampel.

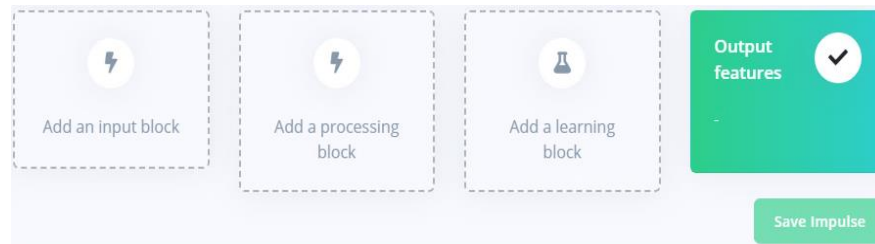
Setelah dataset diunggah, menu *Data Acquisition* menyediakan opsi-opsi CRUD untuk mengatur data-data yang telah diunggah. Pembagian data latih dan uji dilakukan dengan rasio 80:20, data latih berjumlah 6913 sedangkan data uji berjumlah 1711. Gambar 17 menunjukkan tampilan menu ‘Data Acquisition’.



Gambar 17 Laman ‘Data Acquisition’

### 3. Pengaturan impulse

*Impulse* adalah istilah dari Edge Impulse yang mengacu kepada sebuah *pipeline* ML berisi parameter-parameter yang ditentukan oleh pengguna sebelum proses pelatihan model dimulai. Sebuah *Impulse* berisi tiga elemen penyusun utama: blok input (*input block*), blok pemrosesan (*processing block*), dan blok pembelajaran (*learning block*). Gambar 18 menunjukkan tampilan halaman pembuatan *Impulse*.



Gambar 18 Tampilan laman “Create Impulse”

Blok input menunjukkan jenis data masukan untuk pelatihan model. Blok ini terisi secara otomatis berdasarkan dataset yang diunggah. Blok pemrosesan berisi operasi DSP (*Digital Signal Processing*) untuk mengekstraksi fitur. Blok pembelajaran berisi *neural network* untuk mempelajari dataset. Gambar 19 di bawah menampilkan beberapa DSP yang dapat dipilih pada blok pemrosesan dan blok pembelajaran.

DESCRIPTION
<b>Flatten</b> Flatten an axis into a single value, useful for slow-moving averages like temperature data, in combination with other blocks.
<b>Spectral Analysis</b> Great for analyzing repetitive motion, such as data from accelerometers. Extracts the frequency and power characteristics of a signal over time.
<b>Spectrogram</b> Extracts a spectrogram from audio or sensor data, great for non-voice audio or data with continuous frequencies.
<b>IMU (Syntiant)</b> Syntiant only. Great for analyzing repetitive motion, such as data from accelerometers. Extracts the frequency and power characteristics of a signal over time.
<b>Raw Data</b> Use data without pre-processing. Useful if you

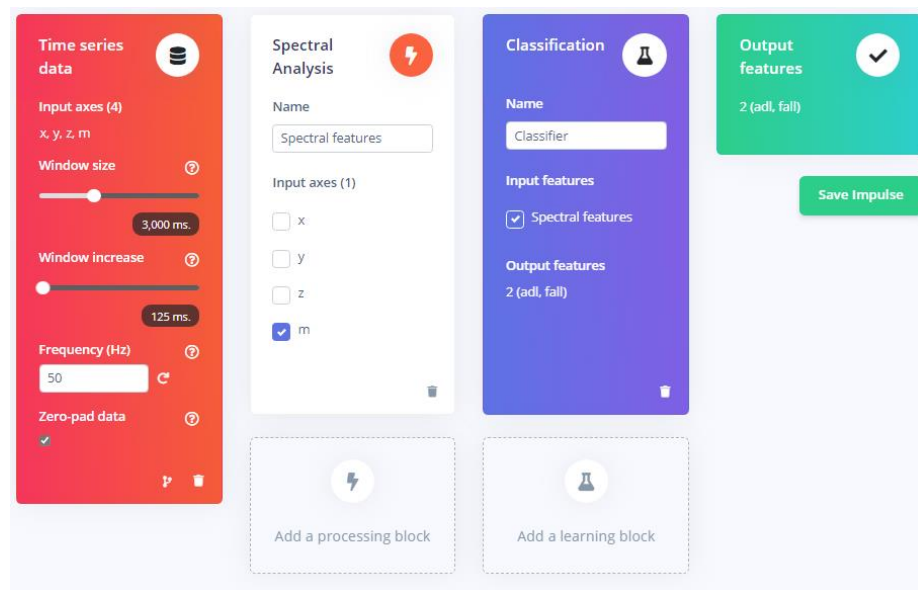
(a)

DESCRIPTION
<b>Classification</b> Learns patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio.
<b>Regression</b> Learns patterns from data, and can apply these to new data. Great for predicting numeric continuous values.
<b>Anomaly Detection (GMM)</b> <span>ENTERPRISE</span> Find outliers in new data. A Gaussian mixture model (GMM) models the shape of data using a probability distribution. New data that is unlikely according to this model can be considered anomalous.
<b>Anomaly Detection (K-means)</b> Find outliers in new data. Good for recognizing unknown states, and to complement classifiers. Works best with low dimensionality features like the output of the spectral features block.
<b>Classification - BrainChip Akida™</b> Learns patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio. Only works with BrainChip AKD1000 MINI PCIe board.

(b)

Gambar 19 Pilihan pada (a) blok pemrosesan dan (b) blok pembelajaran

Pada penelitian ini, DSP yang digunakan adalah *spectral analysis*, sedangkan *neural network* yang digunakan adalah *classifier* Keras. Edge Impulse akan membentuk kedua blok yang telah dipilih menjadi sub-menu di dalam menu ‘Impulse Design’.



Gambar 20 Tampilan laman “Impulse Design”

Gambar 20 menunjukkan tampilan pada sub-menu “Impulse Design” yang berisi blok input, blok pemrosesan, dan blok pembelajaran yang telah terisi. “Input axes” menunjukkan semua sumbu pada dataset, kolom “Window size” berisi ukuran fitur mentah yang digunakan untuk pelatihan model, kolom “Window increase” digunakan untuk menciptakan lebih banyak fitur secara artifisial dan memberikan lebih banyak informasi kepada blok pembelajaran, frekuensi secara otomatis diisi sesuai dataset yang telah diunggah, dan “zero-pad data” bila dicentang akan mengisi atau mengganti fitur mentah yang hilang dengan nilai 0. Blok input merupakan “Time series data”, dimana “window size” data yang digunakan adalah 3000 ms, frekuensi data adalah 50 Hz, dan “zero-pad data” dicentang. Blok pemrosesan merupakan “Spectral Analysis” dan data yang digunakan hanya data magnitudo atau “m”. Blok pembelajaran berupa “Classification” menggunakan Keras.

Selanjutnya pada sub-menu “Spectral Features” terdapat parameter-parameter yang secara otomatis terisi oleh Edge Impulse menggunakan AutoML untuk memudahkan pengguna dalam membangun model berdasarkan dataset dan perangkat target. Namun, nilai-nilai tersebut juga

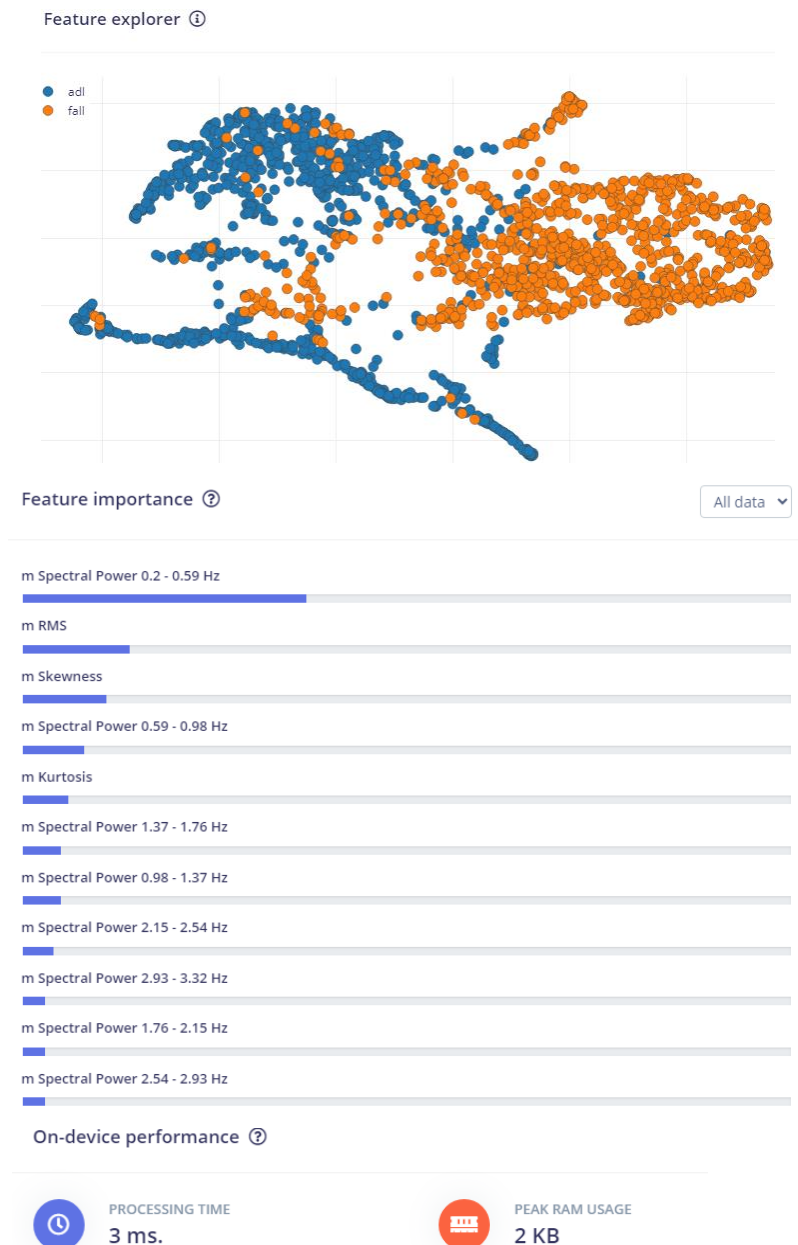
dapat diatur oleh pengguna. Gambar 21 menunjukkan Pengaturan “Spectral Features” yang diimplementasikan pada penelitian ini.

Filter	
Scale axes ?	1
Type ?	low
Cut-off frequency ?	3
Order ?	6
Analysis	
FFT length ?	128
Take log of spectrum? ?	<input checked="" type="checkbox"/>
Overlap FFT frames? ?	<input checked="" type="checkbox"/>

Gambar 21 Pengaturan *Spectral Features*

Pada gambar di atas, terdapat parameter-parameter filter dan analisis. Pada parameter filter terdapat kolom *scale axes*, *type*, *cut-off frequency*, dan *order*. Pengguna dapat mengisi kolom *Scale axes* untuk dikali dengan semua nilai masukan mentah. *Type* merujuk kepada tipe filter yang akan digunakan pada data mentah, yaitu *low-pass*, *high-pass*, atau *none*. Pengguna juga dapat menentukan frekuensi *cut-off* dalam hertz. Kolom *order* merujuk kepada orde filter Butterworth yang dapat diisi dengan angka nol bila tidak ingin sinyal melewati filter. Data yang telah melewati filter akan digunakan untuk mengkomputasi fitur spektral menggunakan Fast Fourier Transform (FFT). Bila opsi ‘take log of spectrum?’ dipilih, log (basis 10) akan diterapkan ke setiap bin FFT. Bila opsi ‘Overlap FFT frames?’ dipilih, *frame-frame* yang berurutan (*sub-window*) akan dapat saling tumpang tindih di dalam window yang lebih besar.

Pengaturan yang diisi tidak banyak berubah dari rekomendasi AutoML. *Scale axes* sejumlah 1, tipe *filter* merupakan *low pass*, frekuensi *cut off* diatur pada 3 hertz pada filter Butterworth orde ke-6. Panjang FFT yaitu 128. Hasil ekstraksi fitur ditampilkan pada gambar berikut.



Gambar 22 Hasil ekstraksi fitur

Gambar 22 menunjukkan sebuah grafik *dot plot* persebaran data-data adl dan fall berdasarkan fitur-fitur yang telah diekstraksi beserta prediksi waktu eksekusi dan penggunaan RAM bagi model.

Selanjutnya, tahap terakhir dalam pengaturan pelatihan model yaitu pengaturan blok pembelajaran. Sub-menu “Classifier” mengandung parameter-parameter pelatihan dan pengaturan arsitektur *neural network* yang telah diisi oleh AutoML dan dapat diatur oleh pengguna. Pada tahap ini pengguna menentukan perangkat target, yaitu perangkat yang akan

menjalankan model yang telah dilatih. Pengaturan lebih lanjut yang diimplementasikan pada penelitian ini ditampilkan pada Gambar 23.

#1 ▾ model #30 ★ Primary version Target: Espressif ESP-EYE (ESP32 240MHz)

### Neural Network settings

#### Training settings

Number of training cycles ②

Use learned optimizer ② ☐

Learning rate ②

Training processor ②

#### Advanced training settings

Validation set size ②  %

Split train/validation set on metadata key ②

Batch size ②

Auto-weight classes ② ☒

Profile int8 model ② ☒

#### Neural network architecture

Input layer (11 features)

Dense layer (20 neurons)

Dense layer (10 neurons)

Add an extra layer

Output layer (2 classes)

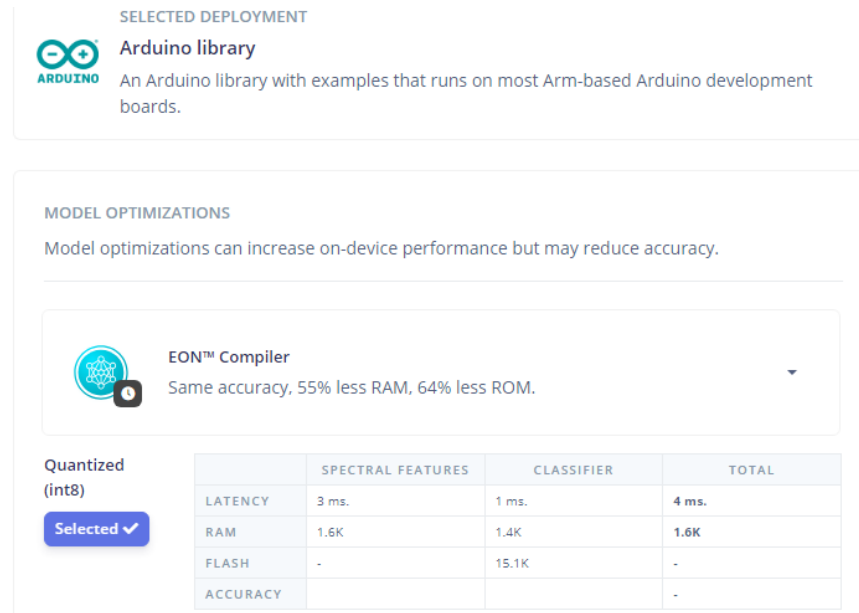
Gambar 23 Pengaturan *Classifier*

*Training cycles (epoch)* diatur sejumlah 100 dan *batch size* berjumlah 64. Jumlah *layer* dan neuron serta pengaturan lain tidak diubah dari rekomendasi AutoML. Setelah pengaturan selesai dilakukan, pengguna melanjutkan dengan menjalankan *classifier* untuk melatih model lalu mengunduh model yang telah dibangun.

#### 4. Deploy model ke mikrokontroler

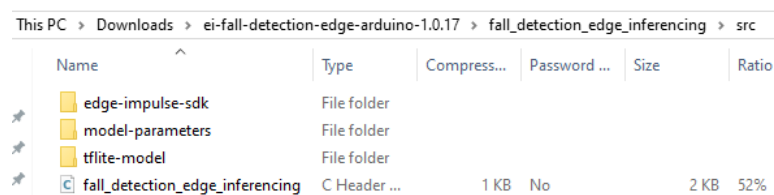
Pada tahap ini, pengguna dapat mengunduh model dalam bentuk file .zip untuk dijalankan sebagai *library*. SDK Edge Impulse dan *library C++*

memungkinkan model yang telah dibangun untuk dijalankan tanpa koneksi internet pada perangkat apapun. Di menu “Deployment”, pengguna dapat mengkonfigurasi jenis unduhan sesuai bahasa pemrograman, menentukan optimasi model, dan jenis kompiler. Gambar 24 menunjukkan pengaturan deployment yang digunakan pada penelitian ini.



Gambar 24 Pengaturan *Deployment*

Pada penelitian ini, model di-*deploy* sebagai sebuah *library* Arduino yang di-*compile* menggunakan kompiler EON, dan dikuantisasi. Hasil *deployment* merupakan sebuah file ZIP untuk digunakan pada program yang mengandung model Edge Impulse, ditampilkan pada gambar 25.

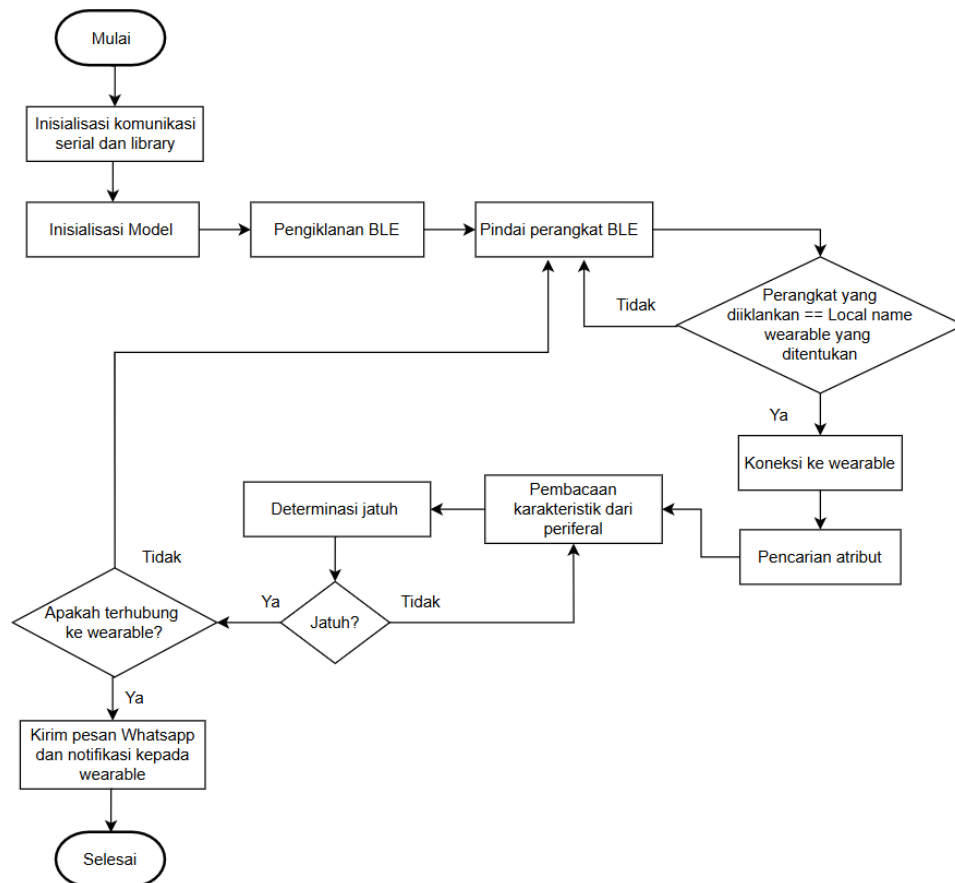
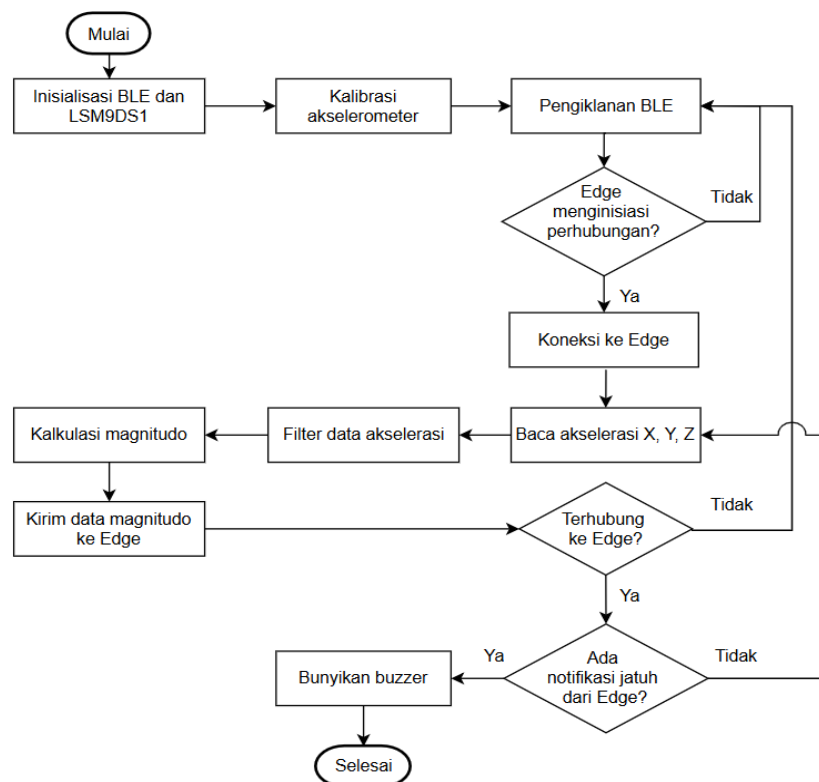


Gambar 25 Isi file ZIP *deployment* model edge impulse

### 3.4.3 Rancangan Software

Program yang dijalankan oleh mikrokontroller edge dibuat menggunakan bahasa C++, sedangkan laptop menjalankan program Python untuk terhubung ke Whatsapp. Sedangkan program yang dijalankan oleh perangkat wearable merupakan sebuah program berbahasa C++.



Gambar 26 *Flowchart* sistem edgeGambar 27 *Flowchart* sistem wearable

Gambar 26 menjelaskan alur kerja *software* edge. Program menginisialisasi komunikasi serial untuk menerima data masukan dari Arduino yang terhubung melalui port USB. Model lalu diinisialisasi sebelum program mulai memindai perangkat BLE di sekitarnya. Setelah kedua perangkat saling terhubung, edge mencari *service* sesuai UUID lalu membaca *characteristic* yang dibagikan. Model akan membuat keputusan berdasarkan nilai-nilai tersebut dan akan memberi notifikasi kepada wearable bila model menghasilkan prediksi jatuh. *Flowchart* alur kerja program wearable ada pada gambar 27.

Pada kedua diagram di atas, perhubungan antara perangkat edge dan wearable melalui BLE dilakukan dengan cara pengiklanan *local name*. Edge menginisiasi koneksi dengan wearable dengan cara memindai *local name* yang telah ditetapkan di dalam program sehingga memungkinkan perhubungan secara otomatis. Berikut gambar *source code* dari program edge dan wearable yang memungkinkan fitur tersebut.

```
#include <ArduinoBLE.h>

void setup() {
  Serial.begin(9600);
  while (!Serial);
}

void loop() {
  BLE.scan(); // pemindaian
  BLEDevice peripheral = BLE.available();
  if (peripheral) {
    if(peripheral.localName() == "peripheral nano") { // nama yang dicari
      BLE.stopScan();
      if (peripheral.connect()) {
        Serial.println("Peripheral connected");
      }
      else {
        Serial.println("Failed to connect!");
        return;
      }
    }
  }
}
```

Gambar 28 *Source code* konektivitas BLE edge

```

#include <ArduinoBLE.h>

void setup() {
  Serial.begin(9600);
  if (!BLE.begin()) {
    Serial.println("BLE failed to Initiate");
    while (1);
  }
  BLE.setLocalName("peripheral nano");          // nama yang diiklankan
  BLE.advertise();                               // pengiklanan
}

void loop() {
  BLEDevice central = BLE.central();
  if (central) {
    if (central.connect()) {
      Serial.println("Connected");
    }
    else {
      Serial.println("Failed to connect!");
      return;
    }
  }
  Serial.print("Disconnected from central: ");
  Serial.println(central.address());
  BLE.advertise();                               // pengiklanan ulang
}

```

Gambar 29 *Source code* konektivitas BLE wearable

Potongan-potongan program yang ditampilkan pada gambar 28 dan 39 memungkinkan edge dan wearable untuk tetap melakukan perhubungan secara otomatis bila selama kedua perangkat hidup belum terjadi perhubungan. Setelah perhubungan terjadi, wearable akan mulai mengakuisisi data akselerasi. Data akselerasi mentah yang didapatkan akan melewati filter *low-pass* untuk menyamakan data latih. Data yang telah difilter digunakan untuk menghasilkan nilai magnitudo yang kemudian dikirimkan ke edge. Nilai-nilai magnitudo yang dikirim secara berkala ke edge ditampung dalam sebuah *array* hingga berjumlah 150 nilai atau setara dengan tiga detik data, lalu diteruskan ke model untuk dijalankan inferensi. Prediksi jatuh yang dihasilkan mengaktifkan fungsi *notify* ke wearable untuk membunyikan buzzer dan mengirim pesan ke Whatsapp kepada nomor yang telah ditentukan. Semua hal tersebut dilakukan sementara fungsi koneksi BLE memeriksa keadaan perhubungan secara berkala.

### 3.5 Evaluasi Model

Performa model diukur dengan melakukan pengujian nyata sambil merekam hasil prediksi model beserta nilai aktualnya, kemudian menerapkan perhitungan akurasi. Subjek pengujian yang merupakan tiga orang dengan rentang usia 20 hingga 30 tahun dan tinggi badan sekitar 155 cm hingga 170 cm mengenakan wearable dalam kantong celana lalu melakukan masing-masing 50 simulasi jatuh dan 50 simulasi ADL dengan durasi 3 detik untuk tiap aktivitas seperti yang dijabarkan pada Tabel 2.

Tabel 2 Spesifikasi subjek pengujian

Subjek	Usia	Tinggi badan (cm)
1	26	155
2	24	160
3	25	170

Tiap aktivitas yang dilakukan akan dibandingkan dengan hasil prediksi model. Metrik yang digunakan untuk mengukur performa model adalah perhitungan akurasi. Akurasi mengukur ketepatan prediksi *classifier* dibandingkan dengan seluruh data uji. Rumus akurasi adalah sebagai berikut,

$$Akurasi = \frac{Jumlah\ Prediksi\ Benar}{Total\ Data}$$

Secara rinci dijabarkan sebagai berikut,

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

Dimana:

*True Positive* (TP): ada kejadian jatuh dan model mendeteksi jatuh.

*True Negative* (TN): tidak ada kejadian jatuh dan model tidak mendeteksi jatuh.

*False Positive* (FP): tidak ada kejadian jatuh, namun model mendeteksi jatuh.

*False Negative* (FN): model tidak berhasil mendeteksi jatuh yang terjadi.

### 3.6 Evaluasi Sistem

Pengujian fungsionalitas sistem bertujuan untuk memastikan bahwa integrasi satu komponen ke komponen lainnya berhasil. Pengujian ini dari segi komunikasi BLE dan keberhasilan edge berkomunikasi dengan Whatsapp. Tabel 3 meringkas aspek-aspek pengujian sistem.

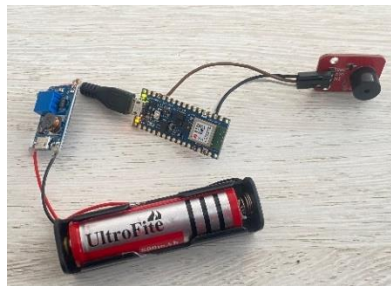
Tabel 3 Skema pengujian sistem

Aspek	Proses	Hasil yang diharapkan
Bluetooth	1. Perhubungan antar perangkat	Edge dan wearable saling terhubung
	2. <i>Reconnect</i> otomatis	Edge dan wearable saling terhubung secara otomatis setelah koneksi sebelumnya gagal atau terputus
	3. Komunikasi wearable ke edge	Edge menerima data akselerasi oleh wearable
	4. Komunikasi edge ke wearable	Wearable menerima notifikasi dari edge ketika prediksi jatuh dihasilkan
Cloud (WA)	1. Terjadi jatuh	Pesan Whatsapp dikirim ke kontak darurat
	2. ADL	Tidak ada pesan Whatsapp yang dikirim
Buzzer	1. Terjadi jatuh	Buzzer berbunyi
	2. ADL	Buzzer tidak berbunyi

## BAB 4 HASIL DAN PEMBAHASAN

### 4.1 Hasil Rancangan Sistem

Berdasarkan rancangan sistem yang telah dijelaskan pada halaman 24, telah dihasilkan sebuah wearable seperti yang ditampilkan pada gambar 30 di bawah ini. Secara keseluruhan, rangkaian tersebut dapat dimuat dalam sebuah *casing* berukuran panjang 9,5 cm, lebar 6 cm, dan tinggi 2 cm.



(a)



(b)



(c)

Gambar 30 Tampak rangkaian *wearable* (a) di luar *casing*, (b) di dalam *casing* tanpa tutup, dan (c) di dalam *casing* dengan tutup.

Edge menginisiasi koneksi dengan wearable dengan cara memindai *local name* yang telah ditetapkan di dalam program sehingga memungkinkan perhubungan secara otomatis bila selama kedua perangkat hidup belum terjadi perhubungan. Gambar 31 dan 32 menampilkan keluaran pada serial monitor edge selama proses pemindaian dan perhubungan.

```

Arduino central device is now active, scanning for peripheral..
Scanning for peripheral..
Scanning for peripheral..
Found 3b:61:33:99:11:24 ''
Scanning for peripheral..
Found 3b:61:33:99:11:24 ''
Scanning for peripheral..
Found 85:3a:49:c9:ee:cd 'peripheral nano' 1101
Connected
Discovering attributes ...
Attributes discovered

Device name: Arduino
Appearance: 0x0

Prediksi total jatuh: 0, adl: 50
Prediksi total jatuh: 0, adl: 50
Prediksi total jatuh: 5, adl: 45
Arduino central device is now active, scanning for peripheral..

Found 1d:aa:cc:11:d0:eb ''
Found 3b:61:33:99:11:24 ''
Found 85:3a:49:c9:ee:cd 'peripheral nano' 1101
Connected
Discovering attributes ...
Attributes discovered

```

Pemindaian  
 Periferal ditemukan, inisiasi perhubungan  
 Penerimaan data  
 Pemindaian ulang  
 Reconnect

Gambar 31 Perhubungan edge

Gambar 32 menampilkan keluaran pada serial monitor wearable selama proses pengiklanan dan perhubungan.

```

Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Connected to central: * Device MAC address: 5c:94:4e:50:d6:6b

Connected
Discovering attributes ...
Attributes discovered
x: -0.01 y: -0.33 z: 0.47 magnitude: 57
x: -0.01 y: -0.39 z: 0.56 magnitude: 68
x: -0.01 y: -0.40 z: 0.58 magnitude: 70
x: -0.01 y: -0.43 z: 0.59 magnitude: 72
x: -0.01 y: -0.41 z: 0.59 magnitude: 71
x: -0.02 y: -0.06 z: 0.64 magnitude: 64
x: -0.03 y: -0.06 z: 0.63 magnitude: 63
x: -0.03 y: -0.06 z: 0.64 magnitude: 64
x: -0.04 y: -0.06 z: 0.64 magnitude: 64
x: -0.03 y: -0.04 z: 0.65 magnitude: 65
x: -0.04 y: -0.05 z: 0.66 magnitude: 66
Disconnected from central: 5c:94:4e:50:d6:6b
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Disconnected from central: 00:00:00:00:00:00
Connected to central: * Device MAC address: 5c:94:4e:50:d6:6b

Connected
Discovering attributes ...
Attributes discovered
x: -0.05 y: -0.74 z: -0.57 magnitude: 93
x: -0.05 y: -0.88 z: -0.82 magnitude: 120
x: -0.05 y: -0.91 z: -0.87 magnitude: 125

```

Status selama pengiklanan  
 Perhubungan sukses  
 Pengiriman data  
 Pengiklanan ulang  
 Reconnect  
 Pengiriman data

Gambar 32 Perhubungan wearable

Bila koneksi antar perangkat telah terbentuk, maka wearable akan mengirim data magnitudo kepada edge. Akselerometer mengakuisisi data tiga dimensi dengan berbagai *noise* yang dapat mengakibatkan penurunan akurasi prediksi sehingga data-data akselerasi tersebut harus diperhalus dengan filter. Gambar 33 menunjukkan keluaran serial monitor wearable berisi data-data akselerasi yang telah difilter. Nilai-nilai akselerasi yang telah melewati filter *low-pass* cenderung rata mendekati nol dan relatif stabil terhadap gejolak. Gambar 33 menunjukkan nilai-nilai akselerasi beserta magnitudonya dari serial monitor wearable.

```
x: -0.83 y: -0.97 z: 0.21 magnitude: 129
x: -1.07 y: -1.23 z: 0.19 magnitude: 164
x: -0.99 y: -1.27 z: 0.10 magnitude: 161
x: -0.83 y: -1.18 z: -0.10 magnitude: 144
x: -0.70 y: -0.92 z: -0.20 magnitude: 117
x: -0.50 y: -0.68 z: -0.24 magnitude: 88
```

Gambar 33 Nilai-nilai akselerasi beserta magnitudonya

Sebagaimana yang telah disebutkan pada halaman 39, ketika edge telah menerima 150 data magnitudo atau setara dengan tiga detik data akselerasi, inferensi akan dilakukan. Bila keputusan jatuh dibuat, edge akan mengirimkan notifikasi jatuh kepada wearable. Namun untuk keperluan *debugging*, wearable juga akan menampilkan notifikasi jatuh yang diterima pada serial monitor. Gambar 34 menampilkan keluaran serial monitor yang menunjukkan: (1) akselerasi pada kegiatan tertentu, (2) bila model berhasil membuat keputusan yang sesuai, dan (3) bila edge berhasil mengirimkan notifikasi kepada wearable.

```
x: 0.13 y: -0.33 z: -1.33 magnitude: 137
x: 0.13 y: -0.37 z: -1.31 magnitude: 136
x: 0.14 y: -0.42 z: -1.28 magnitude: 135
x: 0.16 y: -0.49 z: -1.26 magnitude: 135
x: 0.18 y: -0.55 z: -1.23 magnitude: 136
x: 0.18 y: -0.60 z: -1.18 magnitude: 133
x: 0.18 y: -0.68 z: -1.09 magnitude: 130
x: 0.20 y: -0.79 z: -1.01 magnitude: 129
x: 0.19 y: -0.87 z: -0.84 magnitude: 123
x: 0.21 y: -0.94 z: -0.63 magnitude: 114
x: 0.21 y: -0.92 z: -0.39 magnitude: 102
x: 0.23 y: -0.93 z: -0.21 magnitude: 98
x: 0.23 y: -0.92 z: -0.11 magnitude: 95
x: 0.24 y: -0.90 z: -0.02 magnitude: 93
x: 0.21 y: -0.86 z: 0.00 magnitude: 88
x: 0.23 y: -0.83 z: 0.06 magnitude: 85
x: 0.27 y: -0.75 z: 0.18 magnitude: 81 Fall
x: 0.28 y: -0.69 z: 0.23 magnitude: 78 Fall
x: 0.30 y: -0.62 z: 0.31 magnitude: 75 Fall
x: 0.29 y: -0.50 z: 0.37 magnitude: 68 Fall
x: 0.32 y: -0.36 z: 0.44 magnitude: 65 Fall
x: 0.30 y: -0.21 z: 0.48 magnitude: 60 Fall
```

Array X

Array X+1

Hasil prediksi X

Gambar 34 Notifikasi jatuh dari edge



```

x: 0.04 y: 0.93 z: 0.05 magnitude: 92
x: 0.04 y: 0.93 z: 0.05 magnitude: 92
x: 0.04 y: 0.93 z: 0.05 magnitude: 92
x: 0.04 y: 0.93 z: 0.05 magnitude: 93 Fall
x: 0.03 y: 0.93 z: 0.05 magnitude: 92 Fall

```

Gambar 35 Hasil observasi *threshold* model

Gambar 35 menampilkan keluaran serial monitor untuk mengetahui *threshold* model dalam mendefinisikan jatuh. Ketika akselerometer berada pada posisi dimana magnitudonya mencapai 92, model memprediksi adanya jatuh. Sehingga, dapat dikatakan bahwa *threshold* bagi model untuk mendefinisikan jatuh berada pada angka 92 g.

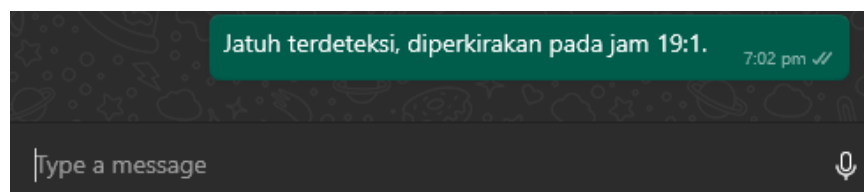
Selain mengirimkan notifikasi kepada wearable, keluaran pada serial monitor edge akan memicu program python untuk mengirimkan pesan darurat melalui whatsapp. Gambar 36 dan 37 menampilkan keluaran pada *command prompt* dan Whatsapp bila jatuh terdeteksi.

```

D:\wapy>python test3.py
total prediksi fall:150, adl: 0
Deteksi Jatuh
In 30 Seconds WhatsApp will open and after 15 Seconds Message will be Delivered!

```

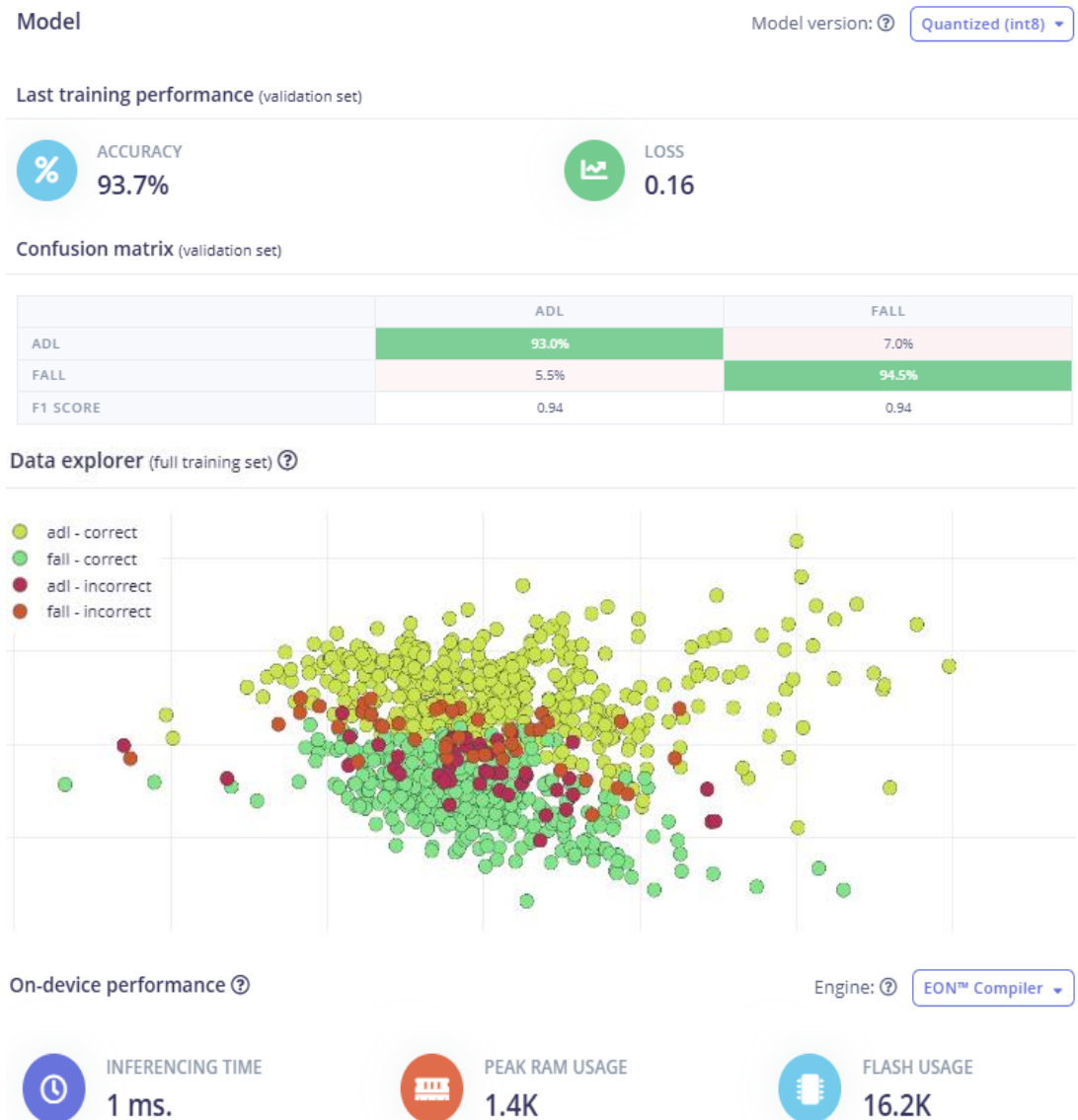
Gambar 36 Keluaran *command prompt* ketika terdeteksi adanya jatuh



Gambar 37 Pesan Whatsapp yang dikirimkan

## 4.2 Evaluasi Model

Model yang dihasilkan dilatih menggunakan *epoch* sejumlah 100, *learning rate* sejumlah 0,0001 dan *batch size* berupa 64. Pada arsitektur *neural network*, *layer* input terdiri atas 11 neuron, terdapat 2 *hidden layer* dengan 20 neuron dan 10 neuron masing-masing, *layer* output mengembalikan 2 kelas.



Gambar 38 Hasil pelatihan model

Gambar 38 menunjukkan sebuah model dengan akurasi sebesar 93,7 % dan *data loss* sebesar 0,16. Grafik *dot plot* menunjukkan persebaran hasil prediksi benar dan salah oleh model. Selain itu, berdasarkan pengaturan *classifier* sebagaimana yang telah dijabarkan pada halaman 35, Edge Impulse memprediksi bahwa model membutuhkan waktu inferensi sebesar 1 ms menggunakan maksimum 1,4 kilobyte RAM dan setidaknya 16,2 kilobyte penyimpanan flash.

Model dapat membedakan aktivitas sehari-hari berintensitas tinggi (lari, lompat) dengan kejadian jatuh. Berikut ini ditampilkan keluaran serial monitor wearable yang menunjukkan akselerasi ketika berlari, melompat, dan jatuh.

```

10:34:50.918 -> x: -0.65 y: -0.58 z: 0.24 magnitude: 90
10:34:51.011 -> x: -0.65 y: -0.60 z: 0.23 magnitude: 91
10:34:51.087 -> x: -0.66 y: -0.61 z: 0.22 magnitude: 92
10:34:51.119 -> x: -0.65 y: -0.60 z: 0.22 magnitude: 91
10:34:51.161 -> x: -0.64 y: -0.61 z: 0.22 magnitude: 91
10:34:51.227 -> x: -0.64 y: -0.61 z: 0.22 magnitude: 91
10:34:51.271 -> x: -0.63 y: -0.61 z: 0.22 magnitude: 90
10:34:51.351 -> x: -0.62 y: -0.61 z: 0.23 magnitude: 90
10:34:51.385 -> x: -0.61 y: -0.61 z: 0.25 magnitude: 89
10:34:51.462 -> x: -0.60 y: -0.61 z: 0.26 magnitude: 89
10:34:51.494 -> x: -0.60 y: -0.61 z: 0.25 magnitude: 88
10:34:51.575 -> x: -0.60 y: -0.61 z: 0.26 magnitude: 88
10:34:51.650 -> x: -0.61 y: -0.61 z: 0.25 magnitude: 89
10:34:51.727 -> x: -0.61 y: -0.61 z: 0.25 magnitude: 89
10:34:51.820 -> x: -0.62 y: -0.61 z: 0.24 magnitude: 89
10:34:51.866 -> x: -0.62 y: -0.61 z: 0.24 magnitude: 90
10:34:51.912 -> x: -0.62 y: -0.61 z: 0.24 magnitude: 90
10:34:51.959 -> x: -0.61 y: -0.60 z: 0.24 magnitude: 89
10:34:51.999 -> x: -0.61 y: -0.60 z: 0.25 magnitude: 89
10:34:52.068 -> x: -0.61 y: -0.61 z: 0.25 magnitude: 89
10:34:52.106 -> x: -0.61 y: -0.60 z: 0.25 magnitude: 89
10:34:52.177 -> x: -0.62 y: -0.60 z: 0.25 magnitude: 89
10:34:52.219 -> x: -0.62 y: -0.60 z: 0.25 magnitude: 89
10:34:52.286 -> x: -0.62 y: -0.60 z: 0.25 magnitude: 89
10:34:52.321 -> x: -0.62 y: -0.60 z: 0.24 magnitude: 90
10:34:52.409 -> x: -0.62 y: -0.60 z: 0.25 magnitude: 90
10:34:52.441 -> x: -0.62 y: -0.60 z: 0.25 magnitude: 89
10:34:52.482 -> x: -0.62 y: -0.60 z: 0.25 magnitude: 89
10:34:52.553 -> x: -0.61 y: -0.60 z: 0.25 magnitude: 89
10:34:52.628 -> x: -0.61 y: -0.60 z: 0.26 magnitude: 89
10:34:52.662 -> x: -0.61 y: -0.60 z: 0.26 magnitude: 89
10:34:52.737 -> x: -0.61 y: -0.60 z: 0.26 magnitude: 89
10:34:52.769 -> x: -0.61 y: -0.60 z: 0.26 magnitude: 89
10:34:52.846 -> x: -0.61 y: -0.60 z: 0.26 magnitude: 89

```

Gambar 39 Akselerasi ketika berlari ringan

```

10:34:52.846 -> Prediksi total jatuh: 53, adl: 97

```

Gambar 40 Keputusan model untuk kegiatan berlari

Tampak pada gambar 39 bahwa magnitudo ketika berlari relatif stabil dengan intensitas sedang pada rentang 89 g hingga 92 g dari pukul 10:34 detik ke 50 hingga 52. Gambar 40 menunjukkan bahwa model tidak membuat keputusan jatuh berdasarkan rentetan nilai tersebut.

```

10:37:14.260 -> x: -0.23 y: -0.86 z: 2.91 magnitude: 304
10:37:14.295 -> x: -0.15 y: -1.05 z: 1.88 magnitude: 215
10:37:14.368 -> x: -0.08 y: -0.66 z: 0.82 magnitude: 105
10:37:14.408 -> x: -0.08 y: -0.35 z: 0.20 magnitude: 40
10:37:14.445 -> x: -0.05 y: -0.12 z: -0.03 magnitude: 13
10:37:14.509 -> x: -0.04 y: 0.07 z: -0.11 magnitude: 13
10:37:14.541 -> x: -0.06 y: 0.15 z: -0.24 magnitude: 29
10:37:14.618 -> x: -0.06 y: 0.37 z: -0.51 magnitude: 63
10:37:14.710 -> x: -0.19 y: 0.25 z: -0.43 magnitude: 53
10:37:14.758 -> x: -0.39 y: -0.34 z: 0.16 magnitude: 53
10:37:14.821 -> x: -0.50 y: -0.75 z: 2.43 magnitude: 259
10:37:14.868 -> x: -0.39 y: -0.87 z: 3.25 magnitude: 339
10:37:14.915 -> x: -0.22 y: -1.06 z: 1.80 magnitude: 209
10:37:14.951 -> x: -0.13 y: -0.58 z: 0.63 magnitude: 86
10:37:14.990 -> x: -0.08 y: -0.27 z: 0.12 magnitude: 30
10:37:15.070 -> x: -0.02 y: -0.02 z: -0.12 magnitude: 12
10:37:15.116 -> x: -0.04 y: 0.20 z: -0.29 magnitude: 35
10:37:15.153 -> x: 0.02 y: 0.16 z: -0.38 magnitude: 41
10:37:15.239 -> x: -0.01 y: 0.32 z: -0.45 magnitude: 55
10:37:15.318 -> x: -0.18 y: 0.17 z: -0.38 magnitude: 45
10:37:15.352 -> x: -0.53 y: -0.53 z: 1.23 magnitude: 143
10:37:15.425 -> x: -0.50 y: -0.55 z: 3.22 magnitude: 330
10:37:15.470 -> x: -0.20 y: -0.95 z: 1.80 magnitude: 204
10:37:15.548 -> x: -0.16 y: -0.58 z: 0.75 magnitude: 95
10:37:15.595 -> x: -0.04 y: -0.23 z: 0.15 magnitude: 27
10:37:15.638 -> x: -0.07 y: 0.04 z: -0.08 magnitude: 11
10:37:15.673 -> x: -0.03 y: 0.12 z: -0.12 magnitude: 17
10:37:15.711 -> x: -0.04 y: 0.17 z: -0.32 magnitude: 35
10:37:15.796 -> x: 0.07 y: 0.23 z: -0.52 magnitude: 57
10:37:15.875 -> x: -0.09 y: 0.44 z: -0.91 magnitude: 101
10:37:15.922 -> x: -0.57 y: -0.00 z: -0.41 magnitude: 69
10:37:15.957 -> x: -0.87 y: -0.31 z: 2.90 magnitude: 303
10:37:16.032 -> x: -0.67 y: -0.73 z: 3.14 magnitude: 329
10:37:16.077 -> x: -0.36 y: -1.01 z: 1.58 magnitude: 190
10:37:16.111 -> x: -0.16 y: -0.78 z: 0.75 magnitude: 109
10:37:16.186 -> x: -0.07 y: -0.44 z: 0.17 magnitude: 48
10:37:16.265 -> x: -0.03 y: -0.02 z: -0.16 magnitude: 16
10:37:16.303 -> x: -0.04 y: 0.19 z: -0.38 magnitude: 42
10:37:16.374 -> x: 0.03 y: 0.40 z: -0.80 magnitude: 89
10:37:16.454 -> x: -0.49 y: 0.21 z: -0.67 magnitude: 85
10:37:16.533 -> x: -1.14 y: -0.20 z: 2.84 magnitude: 306
10:37:16.600 -> x: -0.80 y: -0.65 z: 2.27 magnitude: 249
10:37:16.680 -> x: -0.40 y: -0.67 z: 0.94 magnitude: 122

```

Gambar 41 Akselerasi ketika melakukan lompatan beruntun

```

10:37:16.830 -> Prediksi total jatuh: 59, adl: 91

```

Gambar 42 Keputusan model untuk serangkaian lompatan

Gambar 41 menunjukkan bahwa ketika pemakai melakukan lompatan beruntun, magnitudo yang dihasilkan relatif bergejolak dengan kesenjangan nilai yang tinggi pada rentang 11 hingga 339 dari pukul 10:37 detik ke 14 hingga 16. Gambar 42 menunjukkan bahwa model tidak membuat keputusan jatuh berdasarkan nilai-nilai tersebut.

```

10:36:40.111 -> x: -0.11 y: -0.45 z: 0.58 magnitude: 74
10:36:40.148 -> x: -0.10 y: -0.46 z: 0.57 magnitude: 73
10:36:40.230 -> x: -0.10 y: -0.44 z: 0.57 magnitude: 72
10:36:40.265 -> x: -0.11 y: -0.43 z: 0.58 magnitude: 73
10:36:40.302 -> x: -0.13 y: -0.43 z: 0.58 magnitude: 73
10:36:40.376 -> x: -0.12 y: -0.43 z: 0.58 magnitude: 72
10:36:40.452 -> x: -0.12 y: -0.44 z: 0.57 magnitude: 72
10:36:40.519 -> x: -0.12 y: -0.42 z: 0.57 magnitude: 72
10:36:40.596 -> x: -0.12 y: -0.42 z: 0.57 magnitude: 72
10:36:40.670 -> x: -0.12 y: -0.43 z: 0.58 magnitude: 72
10:36:40.747 -> x: -0.11 y: -0.43 z: 0.58 magnitude: 72
10:36:40.787 -> x: -0.11 y: -0.43 z: 0.57 magnitude: 72
10:36:40.861 -> x: -0.16 y: -0.43 z: 0.57 magnitude: 73
10:36:40.931 -> x: -0.57 y: -0.45 z: 0.53 magnitude: 89
10:36:40.964 -> x: -1.91 y: -0.27 z: 0.76 magnitude: 206
10:36:41.041 -> x: -2.73 y: 0.02 z: -0.68 magnitude: 281
10:36:41.165 -> x: -0.18 y: -0.19 z: -3.03 magnitude: 304
10:36:41.201 -> x: 1.09 y: 0.14 z: -2.87 magnitude: 307
10:36:41.275 -> x: 2.38 y: 0.48 z: -2.99 magnitude: 385
10:36:41.365 -> x: 0.66 y: 0.70 z: -1.23 magnitude: 156
10:36:41.401 -> x: 0.54 y: 0.24 z: -0.86 magnitude: 104
10:36:41.477 -> x: 0.62 y: 0.13 z: -0.84 magnitude: 104
10:36:41.520 -> x: 0.63 y: 0.07 z: -0.78 magnitude: 100
10:36:41.599 -> x: 0.68 y: 0.12 z: -0.85 magnitude: 109
10:36:41.630 -> x: 0.74 y: 0.13 z: -0.91 magnitude: 117
10:36:41.698 -> x: 0.77 y: 0.12 z: -0.90 magnitude: 119
10:36:41.783 -> x: 0.77 y: 0.11 z: -0.86 magnitude: 116
10:36:41.829 -> x: 0.75 y: 0.15 z: -0.87 magnitude: 116
10:36:41.866 -> x: 0.73 y: 0.16 z: -0.85 magnitude: 112
10:36:41.932 -> x: 0.73 y: 0.14 z: -0.83 magnitude: 111
10:36:42.047 -> x: 0.73 y: 0.11 z: -0.83 magnitude: 110
10:36:42.092 -> x: 0.82 y: 0.08 z: -0.88 magnitude: 120
10:36:42.163 -> x: 0.72 y: 0.13 z: -0.87 magnitude: 113
10:36:42.239 -> x: 0.76 y: 0.10 z: -0.89 magnitude: 117
10:36:42.315 -> x: 0.74 y: 0.12 z: -0.91 magnitude: 117
10:36:42.391 -> x: 0.75 y: 0.14 z: -0.90 magnitude: 118
10:36:42.430 -> x: 0.73 y: 0.11 z: -0.89 magnitude: 115

```

Gambar 43 Akselerasi ketika jatuh

```

10:36:48.827 -> Prediksi total jatuh: 124, adl: 26
10:36:48.827 -> Fall

```

Gambar 44 Keputusan model ketika jatuh

Gambar 43 menunjukkan rentetan magnitudo berintensitas sedang ke tinggi ketika jatuh terjadi. Pemakai sedang dalam posisi berdiri pada pukul 10:36 hingga detik ke 40.931 dan jatuh pada detik ke 40.964. Terlihat perubahan nilai yang tajam pada detik ke 40.931 ke detik berikutnya di 40.964 dimana magnitudo 89 g ke 206 g. Pada detik ke 41.365 dan seterusnya, pemakai dalam keadaan berbaring di lantai. Model membuat keputusan jatuh sebagaimana ditampilkan pada Gambar 44.

Setelah integrasi model dalam sistem berhasil dilakukan, akurasi prediksi model akan diuji secara langsung. Uji performa model dilakukan dengan cara melakukan simulasi ADL dan jatuh sejumlah yang dijabarkan pada tabel berikut sambil mengenakan *wearable* dan membandingkannya dengan aktivitas yang sebenarnya.

Tabel 4 Hasil pengujian pada subjek 1 (155 cm)

Kelas Pengujian	Jumlah	Keputusan model	
		Fall	ADL
Fall	50	48	2
ADL	50	8	42

Tabel 5 Hasil pengujian pada subjek 2 (160 cm)

Kelas Pengujian	Jumlah	Keputusan model	
		Fall	ADL
Fall	50	46	4
ADL	50	2	48

Tabel 6 Hasil pengujian pada subjek 3 (170 cm)

Kelas Pengujian	Jumlah	Keputusan model	
		Fall	ADL
Fall	50	48	2
ADL	50	9	41

Tabel 4, 5, dan 6 menunjukkan bahwa tiap subjek pengujian menyimulasikan 100 aktivitas yang secara spesifik berjumlah 50 kali jatuh dan 50 kali ADL. Pada Tabel 4, dari 50 simulasi jatuh terdapat 48 prediksi benar dan dari 50 simulasi ADL terdapat 42 prediksi benar. Pada Tabel 5, dari 50 simulasi jatuh terdapat 46 prediksi benar dan dari 50 simulasi ADL terdapat 48 prediksi benar. Sedangkan pada Tabel 6, dari 50 simulasi jatuh terdapat 48 prediksi benar dan dari 50 simulasi ADL terdapat 41 prediksi benar. Berdasarkan hasil uji yang telah diuraikan di atas, akurasi model adalah sebagai berikut:

$$Akurasi\ 1 = \frac{48 + 42}{48 + 2 + 42 + 8} \times 100\% = \frac{90}{100} \times 100\% = \mathbf{90\%}$$

$$Akurasi\ 2 = \frac{46 + 48}{46 + 4 + 48 + 2} \times 100\% = \frac{94}{100} \times 100\% = \mathbf{94\%}$$

$$Akurasi\ 3 = \frac{48 + 41}{48 + 2 + 41 + 9} \times 100\% = \frac{89}{100} \times 100\% = \mathbf{89\%}$$

Hasil perhitungan akurasi menunjukkan bahwa keputusan model berdasarkan variasi tinggi badan pada rentang 155 hingga 170 cm tidak terdapat perbedaan signifikan. Rerata akurasi dari pengujian yang dilakukan oleh ketiga subjek yaitu:

$$Akurasi = \frac{90 + 94 + 89}{3} \times 100\% = \frac{273}{3} \times 100\% = \mathbf{91\%}$$

Akurasi model yang didapatkan adalah 91%, kurang lebih setara dengan hasil prediksi akurasi yang diproduksi oleh Edge Impulse yaitu 93,7%.

### 4.3 Evaluasi Sistem

Pengujian fungsional sistem yang telah dilakukan dijabarkan pada Tabel 7.

Tabel 7 Hasil pengujian performa sistem

Aspek	Skenario pengujian	Hasil yang diharapkan	Hasil pengujian
Bluetooth Low Energy	1. Perhubungan antar perangkat	Edge dan wearable saling terhubung	Berhasil
	2. <i>Reconnect</i> otomatis	Edge dan wearable saling terhubung secara setelah koneksi sebelumnya gagal atau terputus	Berhasil
	3. Wearable mengirim data ke edge	Edge menerima data akselerasi oleh wearable	Berhasil
	4. Edge mengirim notifikasi ke wearable	Wearable menerima notifikasi dari edge ketika prediksi jatuh dihasilkan	Berhasil
Cloud (WA)	1. Terjadi jatuh	Pesan Whatsapp dikirim ke kontak darurat	Berhasil
	2. ADL	Tidak ada pesan Whatsapp yang dikirim	Berhasil
Buzzer	1. Terjadi jatuh	Buzzer berbunyi	Berhasil
	2. ADL	Buzzer tidak berbunyi	Berhasil



Waktu yang dibutuhkan oleh wearable untuk mengakuisisi data akselerasi, melakukan filtrasi nilai akselerasi, melakukan komputasi nilai magnitudo dan mengirimkan nilai-nilai tersebut melalui BLE yaitu antara 30 hingga 115 milidetik sebagaimana yang ditampilkan pada Gambar 45.

```

10:34:51.011 -> x: -0.65 y: -0.60 z: 0.23 magnitude: 91
10:34:51.087 -> x: -0.66 y: -0.61 z: 0.22 magnitude: 92
10:34:51.119 -> x: -0.65 y: -0.60 z: 0.22 magnitude: 91
10:34:51.161 -> x: -0.64 y: -0.61 z: 0.22 magnitude: 91
10:34:51.227 -> x: -0.64 y: -0.61 z: 0.22 magnitude: 91
10:34:51.271 -> x: -0.63 y: -0.61 z: 0.22 magnitude: 90
10:34:51.351 -> x: -0.62 y: -0.61 z: 0.23 magnitude: 90
10:34:51.385 -> x: -0.61 y: -0.61 z: 0.25 magnitude: 89
10:34:51.462 -> x: -0.60 y: -0.61 z: 0.26 magnitude: 89
10:34:51.494 -> x: -0.60 y: -0.61 z: 0.25 magnitude: 88
10:34:51.575 -> x: -0.60 y: -0.61 z: 0.26 magnitude: 88
10:34:51.650 -> x: -0.61 y: -0.61 z: 0.25 magnitude: 89

```

Gambar 45 Keluaran *serial monitor* wearable dengan *timestamp*

Tabel 8 Cuplikan interval keluaran serial monitor wearable

No.	Time	Interval
1	10:34:50.918	
2	10:34:51.011	00.093
3	10:34:51.087	00.076
4	10:34:51.119	00.032
5	10:34:51.161	00.042
6	10:34:51.227	00.066
7	10:34:51.271	00.044
8	10:34:51.351	00.080
9	10:34:51.385	00.034
10	10:34:51.462	00.077

Dari tabel 8 di atas, dapat dilihat bahwa rerata rentang waktu tiap keluaran serial monitor bagi wearable sekitar 30 hingga 95 milidetik, angka di atas dari nilai tersebut merupakan *outlier* dan sebagaimana yang ditampilkan pada Lampiran 2, terdapat dua nilai di atas rentang yang telah disebutkan, yaitu 115 dan 124 milidetik. Sehingga *mean* interval keluaran serial monitor wearable yaitu 0.0614321 detik atau setara dengan 61 milidetik.

Sedangkan waktu yang dibutuhkan oleh edge untuk melakukan inferensi, dan berkomunikasi dengan wearable yaitu 8 hingga hingga 15 detik sebagaimana yang ditampilkan pada Gambar 46. Sedangkan waktu yang dibutuhkan untuk mengirimkan pesan darurat yaitu kurang lebih 1 menit sebagaimana yang telah ditunjukkan pada Gambar 47 dan 48.



```

10:36:10.472 -> Prediksi total jatuh: 2, adl: 148
10:36:24.774 -> Prediksi total jatuh: 84, adl: 66
10:36:24.774 -> Fall
10:36:39.408 -> Prediksi total jatuh: 13, adl: 137
10:36:48.827 -> Prediksi total jatuh: 124, adl: 26
10:36:48.827 -> Fall
10:36:58.078 -> Prediksi total jatuh: 42, adl: 108

```

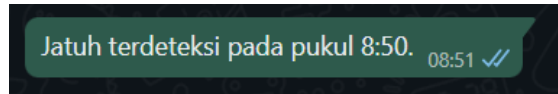
Gambar 46 Keluaran *serial monitor edge* dengan *timestamp*

```

total prediksi Fall: 99, ADL: 51
False
Fall
True
Jatuh terdeteksi
In 45 Seconds WhatsApp will open and after 15 Seconds Message will be Delivered!

```

Gambar 47 Keluaran *command prompt* yang menunjukkan total waktu yang dibutuhkan untuk mengirimkan pesan Whatsapp



Gambar 48 Pesan yang dikirimkan pada 08:51 mengenai jatuh yang terjadi pada pukul 08.50

Tabel 9 Cuplikan interval keluaran serial monitor edge

No.	Time	Interval
1	10:34:43.871	
2	10:34:43.903	00.032
3	10:34:52.846	08.943
4	10:35:01.713	08.867
5	10:35:11.243	09.530
6	10:35:11.289	00.046
7	10:35:20.446	09.157
8	10:35:34.114	13.668
9	10:35:42.947	08.833
10	10:35:51.943	08.996

Tabel 9 menunjukkan cuplikan dari tabel yang secara lengkap dapat dilihat pada Lampiran 3 berisi waktu keluaran serial monitor beserta interval yang didapatkan dari selisih antar waktu. Didapatkan *mean* dari interval keluaran serial monitor edge yaitu 7.93395 detik atau setara dengan 7.9 detik.

## BAB 5 KESIMPULAN DAN SARAN

### 5.1 Kesimpulan

1. Sistem deteksi jatuh pada penelitian ini dicapai menggunakan paradigma *edge computing* dengan pendekatan TinyML serta penggunaan Edge Impulse dalam pelatihan model. Edge Impulse memungkinkan pembangunan model *deep learning* yang dapat dimuat dalam mikrokontroler dengan sumber daya terbatas. Perangkat edge yang kemudian disematkan model tersebut prediksi jatuh berdasarkan data yang disampaikan melalui Bluetooth Low Energy dari perangkat wearable. Teknologi BLE yang digunakan sebagai moda komunikasi antar perangkat terbukti mumpuni dalam transmisi data-data *multi-stream* bervolume rendah secara *real-time* tanpa adanya *data loss* yang terdeteksi. Efektivitas *edge computing* memangkas kebutuhan akan komputasi dan penyimpanan di *cloud*.
2. Model yang dihasilkan terbukti dapat mendeteksi kegiatan sehari-hari dan kejadian jatuh dengan tingkat akurasi 91%, tidak jauh dari prediksi akurasi model yang dibuat oleh Edge Impulse sendiri yaitu sebanyak 93,7%.

### 5.2 Saran

1. Penelitian ini menggunakan sebuah mikrokontroler sebagai perangkat edge sebagai pengimplementasian TinyML, penulis menyarankan menggunakan perangkat yang lebih mumpuni dalam hal kapasitas penyimpanan dan kemampuan pemrosesan sebagai perangkat edge bila jumlah wearable sangat banyak atau data yang ditransmisikan bervolume tinggi.
2. Penelitian berikutnya dapat mengirimkan data agregat ke edge alih-alih mengirimkan data per *sampling rate*.
3. Penulis menyarankan agar menggunakan *library* bleak untuk windows dan bluez untuk linux bila ingin menggunakan BLE pada PC atau SBC.

## DAFTAR PUSTAKA

- Abhishek S. (2021, May 4). *ESP32 Buying Guide*.  
<https://tutorials.probots.co.in/esp32-buying-guide/>
- Alajlan, N. N., & Ibrahim, D. M. (2022). TinyML: Enabling of Inference Deep Learning Models on Ultra-Low-Power IoT Edge Devices for AI Applications. *Micromachines* (Basel).  
<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9227753/>
- Al-Kababji, A., Amira, A., Bensaali, F., Jarouf, A., Shidqi, L., & Djelouat, H. (2021). An IoT-based framework for remote fall monitoring. *Biomedical Signal Processing and Control*, 67, 102532.  
<https://doi.org/10.1016/J.BSPC.2021.102532>
- Alushi, N. (2022). *Accessing Accelerometer Data on Nano 33 BLE*.  
<https://docs.arduino.cc/tutorials/nano-33-ble/imu-accelerometer/>
- Arduino Nano 33 BLE Sense*. (n.d.). Retrieved November 25, 2023, from  
<https://docs.arduino.cc/hardware/nano-33-ble-sense>
- ArduinoBLE*. (n.d.). Retrieved November 26, 2023, from  
<https://www.arduino.cc/reference/en/libraries/arduinoable/>
- Brownlee, J. (2019, August 6). *How to Configure the Learning Rate When Training Deep Learning Neural Networks*.  
<https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/>
- Brownlee, J. (2022, August 15). *Difference Between a Batch and an Epoch in a Neural Network*. <https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/>
- Casilari, E., Santoyo-Ramón, J. A., & Cano-García, J. M. (2020). On the Heterogeneity of Existing Repositories of Movements Intended for the Evaluation of Fall Detection Systems. *Journal of Healthcare Engineering*, 2020. <https://doi.org/10.1155/2020/6622285>
- Edge Computing*. (2019, October 25). <https://nagitec.com/edge-computing/>
- Edge computing vs. cloud computing: What's the difference?* (2021, September 26).  
<https://fauna.com/blog/edge-computing-vs-cloud-computing-whats-the-difference>

- ESP32 Overview*. (n.d.). Retrieved November 25, 2023, from <https://www.espressif.com/en/products/socs/esp32>
- Falls*. (2021, April 26). World Health Organization. <https://www.who.int/news-room/fact-sheets/detail/falls>
- Gjoreski, H., & Gams, M. (2011). *Accelerometer Data Prepration For Activity Recognition*. [https://www.researchgate.net/publication/259340203\\_ACCELEROMETER\\_DATA\\_PREPARATION\\_FOR\\_ACTIVITY\\_RECOGNITION](https://www.researchgate.net/publication/259340203_ACCELEROMETER_DATA_PREPARATION_FOR_ACTIVITY_RECOGNITION)
- Gold, J., & Shaw, K. (2022, June 1). *What is Edge Computing and Why Does It Matter?* Networkworld. <https://www.networkworld.com/article/3224893/what-is-edge-computing-and-how-it-s-changing-the-network.html>
- Grandini, M., Bagli, E., & Visani, G. (2020). *Metrics for Multi-Class Classification: an Overview*. <https://arxiv.org/abs/2008.05756v1>
- Hymel, S., Banbury, C., Situnayake, D., Elium, A., Ward, C., Kelcey, M., Baaijens, M., Majchrzycki, M., Plunkett, J., Tischler, D., Grande, A., Moreau, L., Maslov, D., Beavis, A., Jongboom, J., & Reddi, V. J. (2022). *Edge Impulse: An MLOps Platform for Tiny Machine Learning*. <https://arxiv.org/abs/2212.03332v3>
- Janidarmian, M., Fekr, A. R., Radecka, K., & Zilic, Z. (2017). A Comprehensive Analysis on Wearable Acceleration Sensors in Human Activity Recognition. *Sensors (Basel, Switzerland)*, 17(3). <https://doi.org/10.3390/S17030529>
- Lahouar, S., Mansour, M., & Said, M. H. (2024). *Development of a Fall Detection System Based on a Tri-axial Accelerometer*. <https://www.researchgate.net/publication/377209973>
- Liu, C., Zhang, Y., & Zhou, H. (2021). A comprehensive study of bluetooth low energy. *Journal of Physics: Conference Series*, 2093(1). <https://doi.org/10.1088/1742-6596/2093/1/012021>
- Machine Learning On Arduino Nano 33 BLE Sense*. (2021, March 25). <https://robocraze.com/blogs/post/machine-learning-capabilities-of-arduino-nano-33>

- Mansouri, Y., & Babar, M. A. (2021). A review of edge computing: Features and resource virtualization. *Journal of Parallel and Distributed Computing*, 150, 155–183. <https://doi.org/10.1016/J.JPDC.2020.12.015>
- Micucci, D., Mobilio, M., & Napoletano, P. (2017). UniMiB SHAR: A Dataset for Human Activity Recognition Using Acceleration Data from Smartphones. *Applied Sciences*, 7(10), 1101. <https://doi.org/10.3390/APP7101101>
- Montesinos-López, O. A., Montesinos-López, A., & Crossa Jose. (2022). *Fundamentals of Artificial Neural Networks and Deep Learning*. 379–425. [https://doi.org/DOI: 10.1007/978-3-030-89010-0\\_10](https://doi.org/DOI: 10.1007/978-3-030-89010-0_10)
- Pugh, A., Page, A., & Gabrisko, N. (2019). *Fall Detection and Mitigation*.
- Storr, W. (2013). *Butterworth Filter Design*. [https://www.electronicstutorials.ws/filter/filter\\_8.html](https://www.electronicstutorials.ws/filter/filter_8.html)
- Storr, W. (2013). *Passive Low Pass Filter*. [https://www.electronicstutorials.ws/filter/filter\\_2.html#The\\_Low\\_Pass\\_Filter](https://www.electronicstutorials.ws/filter/filter_2.html#The_Low_Pass_Filter)
- Su, Y. S., & Twu, S. H. (2020). A Real Time Fall Detection System Using Tri-Axial Accelerometer and Clinometer Based on Smart Phones. *IFMBE Proceedings*, 74, 129–137. [https://doi.org/10.1007/978-3-030-30636-6\\_19/COVER](https://doi.org/10.1007/978-3-030-30636-6_19/COVER)
- Thamrin, R. Z., Samijayani, O. N., Rahmatia, S., Adrianto, D., & Enriko, I. K. A. (2020). Implementation of LoRa End-Device in Sensor Network System for Indoor Application. *2020 IEEE International Conference on Communication, Networks and Satellite, Comnetsat 2020 - Proceedings*, 208–212. <https://doi.org/10.1109/COMNETSAT50391.2020.9329003>
- Tomas. (2021, June 26). *TinyML: Machine Learning on ESP32 with MicroPython*. <https://dev.to/tkeyo/tinyml-machine-learning-on-esp32-with-micropython-38a6>
- What is acceleration?* (2015). <https://www.khanacademy.org/science/physics/one-dimensional-motion/acceleration-tutorial/a/acceleration-article>
- What is the Internet of Things?* (2022, August 17). <https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-the-internet-of-things>

- Zhang, X., & Jiang, S. (2021). Application of Fourier Transform and Butterworth Filter in Signal Denoising. *International Conference on Intelligent Computing and Signal Processing (ICSP)*.  
<https://doi.org/10.1109/ICSP51882.2021.9408933>
- Zulkifli, H. (2018, January 22). *Understanding Learning Rates and How It Improves Performance in Deep Learning*.  
<https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10>

### Lampiran 1. *Source code*

*Source code* pada penelitian ini dapat diakses pada link berikut

[saphiranoer/fall\\_detection\\_edge: a fall detection system utilizing edge computing, ble, and edge impulse \(github.com\)](https://saphiranoer/fall_detection_edge: a fall detection system utilizing edge computing, ble, and edge impulse (github.com))

### Lampiran 2. Interval tiap keluaran serial monitor wearable (Sampel 5 detik)

No.	Time	Interval			
1	10:34:50.918		42	10:34:53.272	00.039
2	10:34:51.011	00.093	43	10:34:53.344	00.072
3	10:34:51.087	00.076	44	10:34:53.414	00.070
4	10:34:51.119	00.032	45	10:34:53.491	00.077
5	10:34:51.161	00.042	46	10:34:53.528	00.037
6	10:34:51.227	00.066	47	10:34:53.610	00.082
7	10:34:51.271	00.044	48	10:34:53.645	00.035
8	10:34:51.351	00.080	49	10:34:53.682	00.037
9	10:34:51.385	00.034	50	10:34:53.756	00.074
10	10:34:51.462	00.077	51	10:34:53.832	00.076
11	10:34:51.494	00.032	52	10:34:53.899	00.067
12	10:34:51.575	00.081	53	10:34:53.976	00.077
13	10:34:51.650	00.075	54	10:34:54.050	00.074
14	10:34:51.727	00.077	55	10:34:54.127	00.077
15	10:34:51.820	00.093	56	10:34:54.167	00.040
16	10:34:51.866	00.046	57	10:34:54.241	00.074
17	10:34:51.912	00.046	58	10:34:54.311	00.070
18	10:34:51.959	00.047	59	10:34:54.344	00.033
19	10:34:51.999	00.040	60	10:34:54.421	00.077
20	10:34:52.068	00.069	61	10:34:54.545	00.124
21	10:34:52.106	00.038	62	10:34:54.581	00.036
22	10:34:52.177	00.071	63	10:34:54.655	00.074
23	10:34:52.219	00.042	64	10:34:54.745	00.090
24	10:34:52.286	00.067	65	10:34:54.781	00.036
25	10:34:52.321	00.035	66	10:34:54.857	00.076
26	10:34:52.409	00.088	67	10:34:54.900	00.043
27	10:34:52.441	00.032	68	10:34:54.979	00.079
28	10:34:52.482	00.041	69	10:34:55.010	00.031
29	10:34:52.553	00.071	70	10:34:55.078	00.068
30	10:34:52.628	00.075	71	10:34:55.163	00.085
31	10:34:52.662	00.034	72	10:34:55.209	00.046
32	10:34:52.737	00.075	73	10:34:55.246	00.037
33	10:34:52.769	00.032	74	10:34:55.312	00.066
34	10:34:52.846	00.077	75	10:34:55.427	00.115
35	10:34:52.886	00.040	76	10:34:55.472	00.045
36	10:34:52.971	00.085	77	10:34:55.543	00.071
37	10:34:53.017	00.046	78	10:34:55.619	00.076
38	10:34:53.049	00.032	79	10:34:55.695	00.076
39	10:34:53.122	00.073	80	10:34:55.771	00.076
40	10:34:53.200	00.078	81	10:34:55.810	00.039
41	10:34:53.233	00.033	82	10:34:55.894	00.084

## Lampiran 3. Interval tiap keluaran serial monitor edge (Sampel 30 detik)

No.	Time	Interval
1	10:34:43.871	
2	10:34:43.903	00.032
3	10:34:52.846	08.943
4	10:35:01.713	08.867
5	10:35:11.243	09.530
6	10:35:11.289	00.046
7	10:35:20.446	09.157
8	10:35:34.114	13.668
9	10:35:42.947	08.833
10	10:35:51.943	08.996
11	10:36:00.822	08.879
12	10:36:10.472	09.650
13	10:36:24.774	14.302
14	10:36:39.408	14.634
15	10:36:48.827	09.419
16	10:36:58.078	09.251
17	10:37:07.514	09.436
18	10:37:07.514	00.000
19	10:37:16.830	09.316
20	10:37:25.587	08.757
21	10:37:34.399	08.812
22	10:37:43.223	08.824
23	10:37:54.286	11.063


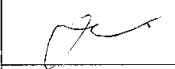




## Lampiran 4. Lembar Perbaikan Skripsi

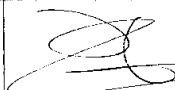
**LEMBAR PERBAIKAN SKRIPSI****“SISTEM WEARABLE PENDETEKSI JATUH DENGAN  
IMPLEMENTASI EDGE COMPUTING”****OLEH:****SAPHIRA NOER S  
D121171520**

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana pada tanggal 01 Juli 2024.  
Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan  
pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr. Adnan, S.T., M.T.	
Sekretaris	Dr. Ir. Zahir Zainuddin, M.Sc.	
Anggota	A. Ais Prayogi Alimuddin, S.T., M.Eng.	
	Muhammad Alief Fadhal Imran Oemar, S.T., M.Sc.	

Persetujuan perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Adnan, S.T., M.T.	
II	Dr. Ir. Zahir Zainuddin, M.Sc.	