



## **TUGAS AKHIR - EC224801**

# **KLASIFIKASI KANTUK BERDASARKAN POLA KEDIPAN MATA dan POLA BUKAAN MULUT dengan IndRNN**

**Adritia Alfiana Merdila**

NRP 0721 19 4000 0017

Dosen Pembimbing

**Ahmad Zaini, S.T., M.Sc**

NIP 19750419 200212 1 003

**Reza Fuad Rachmadi, S.T., M.T., Ph.D.**

NIP 19850403 201212 1 001

**Program Studi Strata 1 (S1) Teknik Komputer**

Departemen Teknik Komputer

Fakultas Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2023



## **TUGAS AKHIR - EC224801**

# **KLASIFIKASI KANTUK BERDASARKAN POLA KEDIPAN MATA dan POLA BUKAAN MULUT dengan IndRNN**

**Adritia Alfiana Merdila**

NRP 0721 19 4000 0017

Dosen Pembimbing

**Ahmad Zaini, S.T., M.Sc**

NIP 19750419 200212 1 003

**Reza Fuad Rachmadi, S.T., M.T., Ph.D.**

NIP 19850403 201212 1 001

**Program Studi Strata 1 (S1) Teknik Komputer**

Departemen Teknik Komputer

Fakultas Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2023

*[Halaman ini sengaja dikosongkan]*



## **FINAL PROJECT - EC224801**

# **DROWSINESS CLASSIFICATION BASED ON EYE BLINK PATTERN AND MOUTH OPENING PATTERN USING *IndRNN***

**Adritia Alfiana Merdila**

NRP 0721 19 4000 0017

Advisor

**Ahmad Zaini, S.T., M.Sc**

NIP 19750419 200212 1 003

**Reza Fuad Rachmadi, S.T., M.T., Ph.D.**

NIP 19850403 201212 1 001

**Undergraduate Study Program of Computer Engineering**

Department of Computer Engineering

Faculty of Faculty of Intelligent Electrical and Informatics Technology

Sepuluh Nopember Institute of Technology

Surabaya

2023

*[Halaman ini sengaja dikosongkan]*

## LEMBAR PENGESAHAN

### KLASIFIKASI KANTUK BERDASARKAN POLA KEDIPAN MATA dan POLA BUKAAN MULUT dengan IndRNN

#### TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat  
memperoleh gelar Sarjana Teknik pada  
Program Studi S-1 Teknik Komputer

Departemen Teknik Komputer  
Fakultas Fakultas Teknologi Elektro dan Informatika Cerdas  
Institut Teknologi Sepuluh Nopember

Oleh: **Adritia Alfiana Merdila**  
NRP. 0721 19 4000 0017

Disetujui oleh Tim Penguji Tugas Akhir:

Ahmad Zaini, S.T., M.Sc  
NIP: 19750419 200212 1 003

(Pembimbing I)



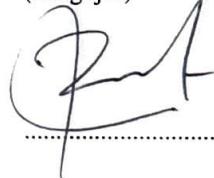
Reza Fuad Rachmadi, S.T., M.T., Ph.D.  
NIP: 19850403 201212 1 001

(Pembimbing II)



Eko Pramunanto, S.T., M.T.  
NIP: 19661203 199412 1 001

(Penguji I)



Dr. Susi Juniastuti, S.T., M.Eng..  
NIP: 19650618 199903 2 001

(Penguji II)



SURABAYA

Juli, 2023

*[Halaman ini sengaja dikosongkan]*

## APPROVAL SHEET

### DROWSINESS CLASSIFICATION BASED ON EYE BLINK PATTERN AND MOUTH OPENING PATTERN USING IndRNN

#### FINAL PROJECT

Submitted to fulfill one of the requirements  
for obtaining a degree Bachelor of Engineering at  
Undergraduate Study Program of Computer Engineering  
Department of Computer Engineering  
Faculty of Faculty of Intelligent Electrical and Informatics Technology  
Sepuluh Nopember Institute of Technology

By: **Adritia Alfiana Merdila**  
NRP. 0721 19 4000 0017

Approved by Final Project Examiner Team:

Ahmad Zaini, S.T., M.Sc  
NIP: 19750419 200212 1 003

(Advisor I)



Reza Fuad Rachmadi, S.T., M.T., Ph.D.  
NIP: 19850403 201212 1 001

(Co-Advisor II)



Eko Pramunanto, S.T., M.T..  
NIP: 19661203 199412 1 001

(Examiner I)



Dr. Susi Juniastuti, S.T., M.Eng..  
NIP: 19650618 199903 2 001

(Examiner II)



Acknowledged,  
Head of Computer Engineering Department F-ELECTICS - ITS



Dr. Supeno Mardi Susiki Nugroho, S.T., M.T..  
NIP. 19700313199512 1 001

SURABAYA

July, 2023

*[Halaman ini sengaja dikosongkan]*

## **PERNYATAAN ORISINALITAS**

Yang bertanda tangan dibawah ini:

Nama Mahasiswa / NRP : Adritia Alfiana Merdila / 0721 19 4000 0017  
Departemen : Teknik Komputer  
Dosen Pembimbing / NIP : Ahmad Zaini, S.T., M.Sc / 19750419 200212 1 003

Dengan ini menyatakan bahwa Tugas Akhir dengan judul "KLASIFIKASI KANTUK BERDASARKAN POLA KEDIPAN MATA dan POLA BUKAAN MULUT dengan IndRNN" adalah hasil karya sendiri, berfsifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, Juli 2023

Mengetahui  
Dosen Pembimbing

  
Ahmad Zaini, S.T., M.Sc  
NIP. 19750419 200212 1 003

Mahasiswa

  
Adritia Alfiana Merdila  
NRP. 0721 19 4000 0017

*[Halaman ini sengaja dikosongkan]*

## **STATEMENT OF ORIGINALITY**

The undersigned below:

Name of student / NRP : Adritia Alfiana Merdila / 0721 19 4000 0017  
Department : Computer Engineering  
Advisor / NIP : Ahmad Zaini, S.T., M.Sc / 19750419 200212 1 003

Hereby declared that the Final Project with the title of "*DROWSINESS CLASSIFICATION BASED ON EYE BLINK PATTERN AND MOUTH OPENING PATTERN USING IndRNN*" is the result of my own work, is original, and is written by following the rules of scientific writing.

If in future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with provisions that apply at Sepuluh Nopember Institute of Technology.

Surabaya, July 2023

Acknowledged

Advisor

Student



Ahmad Zaini, S.T., M.Sc  
NIP. 19750419 200212 1 003



Adritia Alfiana Merdila  
NRP. 0721 19 4000 0017

*[Halaman ini sengaja dikosongkan]*

## **ABSTRAK**

Nama Mahasiswa : Adritia Alfiana Merdila  
Judul Tugas Akhir : KLASIFIKASI KANTUK BERDASARKAN POLA KEDIPAN MATA dan POLA BUKAAN MULUT dengan IndRNN  
Pembimbing : 1. Ahmad Zaini, S.T., M.Sc  
                  2. Reza Fuad Rachmadi, S.T., M.T., Ph.D.

Kantuk adalah keadaan dimana seseorang membutuhkan waktu istirahat dan sering kali muncul pada saat yang tak terduga, seperti saat belajar, bekerja, atau mengemudi. Gejala dari rasa kantuk umumnya adalah microsleeps (berkedip dengan durasi lebih dari 500 ms) dan men-guap. Salah satu cara untuk mendeteksi kantuk yaitu dengan metode non-intrusif menggunakan visi komputer dengan menerapkan algoritma *Deep Learning. Recurrent Neural Network* merupakan salah satu jenis *Deep Learning* yang melibatkan masukan berurutan karena memiliki memori untuk menyimpan *state*. IndRNN merupakan pengembangan dari RNN namun pemrosesan setiap neuron pada suatu layer tidak bergantung dengan layer lain. Melalui penelitian ini model klasifikasi kantuk akan dibuat dengan menggunakan parameter kedipan mata dan pola bukaan mulut dengan menggunakan IndRNN. Penelitian ini mengambil data berupa video dari database UTA-RLDD. Data tersebut kemudian diproses sehingga didapatkan nilai *Eye Aspect Ratio* (EAR) dan *Mouth Aspect Ratio* (MAR) pada setiap frame. EAR dan MAR tersebut nantinya akan digunakan sebagai masukan model klasifikasi pada IndRNN yang menghasilkan output berupa 3 kelas yang mengacu pada skala kantuk Karolinska. Kelas tersebut diantaranya : Kelas pertama (Waspada) yaitu terdiri dari label kelas KSS 1 sampai 3, Kelas kedua (Waspada Rendah) yaitu terdiri dari label kelas KSS 6 dan 7, Kelas ketiga (Mengantuk) yaitu terdiri dari label kelas KSS 8 dan 9. Berdasarkan penelitian yang telah dilakukan, model klasifikasi kantuk yang paling efektif adalah model IndRNN dengan data teraugmentasi. Model tersebut memiliki nilai akurasi sebesar 0,6943 dan niali validasi sebesar 0,52.

Kata Kunci: IndRNN, Kantuk, Visi Komputer.

*[Halaman ini sengaja dikosongkan]*

## ABSTRACT

*Name : Adritia Alfiana Merdila  
Title : DROWSINESS CLASSIFICATION BASED ON EYE BLINK PATTERN AND MOUTH OPENING PATTERN USING IndRNN  
Advisors : 1. Ahmad Zaini, S.T., M.Sc  
2. Reza Fuad Rachmadi, S.T., M.T., Ph.D.*

*Drowsiness is a condition in which a person needs rest and often occurs unexpectedly, such as during studying, working, or driving. Common symptoms of drowsiness are microsleeps (blinks longer than 500 ms) and yawning. One way to detect drowsiness is by using a non-intrusive method using computer vision by applying the Deep Learning algorithm. RNN is a type of Deep Learning that involves sequential input because it has memory to store state. IndRNN is a development of RNN but the processing of each neuron in a layer is independent of other layers. Through this research a drowsiness classification model will be created using the parameters of eye blink and mouth opening pattern using IndRNN. This study retrieved data in the form of videos from the DROZY and UTA-RLDD databases. The data is then processed to obtain the Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR) values for each frame. The EAR and MAR will later be used as input for the classification model in IndRNN which produces output in the form of 3 classes that refer to the Karolinska sleepiness scale. These classes include: First class (Alert) which consists of class labels KSS 1 to 5, Second class (Low Vigilant) which consists of class labels KSS 6 and 7, Third class (Drowsy) which consists of class labels KSS 8 and 9. Based on the conducted research, the most effective drowsiness classification model is the IndRNN model with augmented data. It achieved an accuracy score of 0.6943 and a validation score of 0.52.*

*Keywords: Computer Vision, Drowsiness, IndRNN.*

*[Halaman ini sengaja dikosongkan]*

## KATA PENGANTAR

Puji dan syukur atas kehadiran Allah SWT yang telah melimpahkan rahmat, nikmat, serta hidayah-Nya sehingga skripsi yang berjudul **KLASIFIKASI KANTUK BERDASARKAN POLA KEDIPAN MATA dan POLA BUKAAN MULUT dengan IndRNN** dapat diselesaikan dengan baik.

Penelitian ini disusun dalam rangka memenuhi salah satu syarat kelulusan Program Studi S-1 Teknik Komputer Departemen Teknik Komputer Fakultas Teknologi Elektro dan Informatika Cerdas Institut Teknologi Sepuluh Nopember. Tugas akhir ini mustahil diselesaikan tanpa adanya dukungan dan bimbingan dari berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Bapak Ahmad Zaini, S.T., M.Sc selaku dosen pembimbing I yang telah banyak memberikan semangat, masukan, serta arahan kepada penulis mengenai penelitian dan penulisan buku Tugas Akhir.
2. Bapak Reza Fuad Rachmadi, S.T., M.T., Ph.D. selaku dosen pembimbing II yang telah banyak memberikan masukan serta arahan pada pengerjaan penelitian dan penulisan buku Tugas Akhir.
3. Ibu, Ayah dan Saudara yang selalu mendoakan dan mendukung penulis secara emosional maupun finansial.
4. Teman-teman tercinta yang selalu ada untuk penulis dan senantiasa memberikan semangat kepada penulis.
5. Bapak dan Ibu Dosen Teknik Komputer ITS untuk segala ilmu dan motivasi yang telah diberikan selama perkuliahan.
6. Bapak dan Ibu staff karyawan Teknik Komputer ITS untuk segala usahanya untuk menciptakan lingkungan perkuliahan yang nyaman.

Proposal ini jauh dari kata sempurna. Oleh karena itu penulis mengharapkan kritik dan saran yang dapat digunakan untuk meningkatkan dan mengoreksi penelitian ini. Akhir kata, semoga penelitian yang telah dibuat dapat bermanfaat dan dapat diterima dengan baik oleh banyak pihak.

Surabaya, Juli 2023



Adritia Alfiana Merdila

*[Halaman ini sengaja dikosongkan]*

# **DAFTAR ISI**

<b>ABSTRAK</b>	<b>i</b>
<b>ABSTRACT</b>	<b>iii</b>
<b>KATA PENGANTAR</b>	<b>v</b>
<b>DAFTAR ISI</b>	<b>vii</b>
<b>DAFTAR GAMBAR</b>	<b>xii</b>
<b>DAFTAR TABEL</b>	<b>xiii</b>
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	2
1.3 Tujuan . . . . .	2
1.4 Batasan Masalah . . . . .	2
1.5 Manfaat . . . . .	3
<b>2 TINJAUAN PUSTAKA</b>	<b>5</b>
2.1 Hasil penelitian terdahulu . . . . .	5
2.1.1 Pengembangan Sistem Deteksi Kantuk Menggunakan Pengklasifikasi <i>Random Forest</i> pada Sinyal Elektrokardiogram . . . . .	5
2.1.2 Deteksi Kantuk Berdasarkan Pola Bukaan Mulut Menggunakan Metode 1D CNN . . . . .	5
2.1.3 Klasifikasi Skala Kantuk Karolinska Berdasarkan Nilai <i>Eye Aspect Ratio</i> Menggunakan <i>Deep Learning</i> . . . . .	5
2.1.4 Deteksi Kantuk Pada Pengendara Roda Empat Melalui Citra Wajah Menggunakan Metode <i>Facial Landmark</i> . . . . .	6
2.2 <i>Library Dlib</i> . . . . .	6
2.3 <i>Facial Landmark</i> . . . . .	6
2.3.1 <i>Eye Aspect Ratio (EAR)</i> . . . . .	7
2.3.2 <i>Mouth Aspect Ratio (MAR)</i> . . . . .	8
2.4 Skala Kantuk Karolinska(SKK) . . . . .	8

2.5	Interpolasi Data . . . . .	9
2.6	Normalisasi . . . . .	9
2.7	Augmentasi Data . . . . .	10
2.8	<i>Deep Learning</i> . . . . .	10
2.9	<i>Recurrent Neural Network (RNN)</i> . . . . .	10
2.10	<i>Independent RNN (IndRNN)</i> . . . . .	11
2.11	<i>Long Short Terms Memory (LSTM)</i> . . . . .	12
2.12	<i>Support Vector Machine (SVM)</i> . . . . .	13
<b>3</b>	<b>METODOLOGI</b>	<b>15</b>
3.1	Metode yang digunakan . . . . .	15
3.1.1	Pengumpulan Data . . . . .	16
3.1.2	Ekstraksi Fitur . . . . .	16
3.1.3	<i>Preprocessing Data</i> . . . . .	17
3.1.4	<i>Training Data</i> . . . . .	19
3.1.5	Evaluasi Model . . . . .	23
3.1.6	Pengujian Model . . . . .	24
3.2	Data dan peralatan yang digunakan . . . . .	24
3.2.1	Dataset UTA-RLDD . . . . .	24
3.2.2	Dataset DROZY . . . . .	25
3.2.3	Peralatan . . . . .	25
<b>4</b>	<b>HASIL DAN PEMBAHASAN</b>	<b>27</b>
4.1	Hasil Pengumpulan Data . . . . .	27
4.2	Hasil Ekstraksi Fitur . . . . .	27
4.3	Hasil Preprocessing . . . . .	28
4.3.1	Hasil Interpolasi Data . . . . .	29
4.3.2	Hasil <i>Data Balancing</i> dan <i>Labeling</i> . . . . .	29
4.3.3	Hasil Normalisasi Data . . . . .	30
4.3.4	Hasil Sliding Window . . . . .	31
4.4	Hasil <i>Data Training</i> . . . . .	31
4.4.1	Independent Recurrent Neural Network . . . . .	31
4.4.2	Long short term memory network . . . . .	34
4.5	Hasil Evaluasi Model . . . . .	36
4.5.1	Independent Recurrent Neural Network . . . . .	36

4.5.2	Long short term memory network . . . . .	38
4.5.3	Support Vector Machine . . . . .	40
4.6	Hasil Pengujian Model . . . . .	42
4.6.1	Independent Recurrent Neural Network . . . . .	42
4.6.2	Long short term memory network . . . . .	44
4.6.3	Support Vector Machine . . . . .	46
4.7	Pembahasan . . . . .	48
<b>5</b>	<b>KESIMPULAN DAN SARAN</b>	<b>51</b>
5.1	Kesimpulan . . . . .	51
5.2	Saran . . . . .	51
<b>DAFTAR PUSTAKA</b>		<b>53</b>
<b>LAMPIRAN</b>		<b>55</b>
<b>BIOGRAFI PENULIS</b>		<b>61</b>

*[Halaman ini sengaja dikosongkan]*

## DAFTAR GAMBAR

2.1	Landmark wajah manusia (Ibrahim et al., 2021) . . . . .	6
2.2	enam titik landmark pada mata (Rogalska et al., 2019) . . . . .	7
2.3	Enam titik landmark pada mulut (Sri Mounika et al., 2022) . . . . .	8
2.4	Ilustrasi RNN (LeCun et al., 2015) . . . . .	11
2.5	Ilustrasi IndRNN (Li et al., 2018) . . . . .	11
2.6	Struktur LSTM . . . . .	12
2.7	SVM multiklas . . . . .	13
3.1	Diagram alur model . . . . .	15
3.2	Proses ekstraksi nilai EAR dan MAR . . . . .	16
3.3	Diagram Preprocessing Data . . . . .	17
3.4	Ilustrasi pemotongan data . . . . .	18
3.5	Arsitektur Model Klasifikasi IndRNN . . . . .	20
3.6	Arsitektur Model Klasifikasi LSTM . . . . .	22
3.7	Sampel data UTA-RLDD dalam status waspada (baris pertama), waspada rendah (baris kedua), dan mengantuk (baris ketiga) (Ghoddoosian et al., 2019) . .	24
3.8	Sampel data DROZY (Massoz et al., 2016) . . . . .	25
4.1	Akurasi IndRNN Tanpa Data Augmentasi . . . . .	32
4.2	Loss IndRNN Tanpa Data Augmentasi . . . . .	32
4.3	Akurasi dan Loss IndRNN Dengan Data Augmentasi . . . . .	33
4.4	Akurasi dan Loss IndRNN Dengan Data Augmentasi . . . . .	33
4.5	Akurasi dan Loss LSTM Tanpa Data Augmentasi . . . . .	34
4.6	Akurasi dan Loss LSTM Tanpa Data Augmentasi . . . . .	34
4.7	Akurasi dan Loss LSTM Dengan Data Augmentasi . . . . .	35
4.8	Akurasi dan Loss LSTM Dengan Data Augmentasi . . . . .	35
4.9	Confusion Matrix IndRNN Tanpa Data Augmentasi . . . . .	37
4.10	Evaluasi IndRNN Tanpa Data Augmentasi . . . . .	37
4.11	Confusion Matrix IndRNN Dengan Data Augmentasi . . . . .	38
4.12	Evaluasi IndRNN Dengan Data Augmentasi . . . . .	38
4.13	Akurasi dan Loss LSTM Tanpa Data Augmentasi . . . . .	39
4.14	Evaluasi LSTM Dengan Data Augmentasi . . . . .	39

4.15 Confusion Matrix LSTM Dengan Data Augmentasi . . . . .	40
4.16 Evaluasi LSTM Dengan Data Augmentasi . . . . .	40
4.17 Confusion Matrix SVM Tanpa Data Augmentasi . . . . .	41
4.18 Evaluasi SVM Dengan Data Augmentasi . . . . .	41
4.19 Confusion Matrix SVM Dengan Data Augmentasi . . . . .	42
4.20 Evaluasi SVM Dengan Data Augmentasi . . . . .	42
4.21 Confusion Matrix IndRNN Tanpa Data Augmentasi . . . . .	43
4.22 Testing IndRNN Tanpa Data Augmentasi . . . . .	43
4.23 Confusion Matrix IndRNN Dengan Data Augmentasi . . . . .	44
4.24 Testing IndRNN Dengan Data Augmentasi . . . . .	44
4.25 Confusion Matrix LSTM Tanpa Data Augmentasi . . . . .	45
4.26 Testing LSTM Dengan Data Augmentasi . . . . .	45
4.27 Confusion Matrix LSTM Dengan Data Augmentasi . . . . .	46
4.28 Testing LSTM Dengan Data Augmentasi . . . . .	46
4.29 Confusion Matrix SVM Tanpa Data Augmentasi . . . . .	47
4.30 Testing SVM Dengan Data Augmentasi . . . . .	47
4.31 Confusion Matrix SVM Dengan Data Augmentasi . . . . .	48
4.32 Testing SVM Dengan Data Augmentasi . . . . .	48

## **DAFTAR TABEL**

2.1	Koordinat landmark wajah . . . . .	7
2.2	Skala Kantuk Karolinska (Putilov & Donskaya, 2013) . . . . .	8
3.1	Pengaturan Hyperparameter Model Klasifikasi . . . . .	22
4.1	Sebaran Video Tiap Kelas . . . . .	27
4.2	Deskripsi Data Hasil Ekstraksi Fitur . . . . .	27
4.3	Data Kosong Sebelum Interpolasi . . . . .	29
4.4	Data Kosong Setelah Interpolasi . . . . .	29
4.5	Jumlah Data Pada Tiap Video . . . . .	29
4.6	Data Sebelum Normalisasi . . . . .	31
4.7	Data Setelah Normalisasi . . . . .	31
4.8	Data Hasil Training Model . . . . .	48
4.9	Data Hasil Evaluasi Model . . . . .	49

*[Halaman ini sengaja dikosongkan]*

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Kantuk adalah keadaan di mana seseorang membutuhkan waktu istirahat dan sering kali muncul pada saat yang tak terduga, seperti saat belajar, bekerja, atau mengemudi. Mengantuk dapat menyebabkan gejala yang berdampak besar terhadap kinerja tugas seperti: waktu respons yang melambat, kurangnya kesadaran intermiten, atau *microsleeps* (berkedip dengan durasi lebih dari 500 ms) (Dinges, 1995). Ciri lain dari orang yang mengantuk adalah sering menguap pada jangka waktu tertentu (Dedy Irawan et al., 2019).

Salah satu masalah yang berhubungan dengan K3 (Kesehatan dan Keselamatan Kerja) yang bisa menyebabkan kecelakaan kerja adalah kelelahan. Kelelahan kerja adalah kondisi di mana efisiensi dan ketahanan seseorang dalam bekerja menurun, yang mengakibatkan berkurangnya kemauan untuk bekerja. Gejala yang biasa dialami termasuk perasaan letih di seluruh tubuh, menguap, mengantuk, kesulitan fokus dan berkonsentrasi, dan sebagainya. Dampak dari kelelahan kerja adalah menurunnya kinerja tubuh dan produktivitas. Selain itu, kelelahan kerja juga dapat menyebabkan timbulnya penyakit akibat kerja (PAK) dan kecelakaan akibat pekerjaan(Ardinendradewi et al., 2022).

Mendeteksi kantuk merupakan salah satu cara untuk menghindari dampak negatif dari rasa kantuk itu sendiri. Beberapa cara dapat digunakan untuk mendeteksi rasa kantuk contohnya dengan metode intrusif dimana pendekslsian rasa kantuk dilakukan dengan memanfaatkan bantuan alat EKG (*Electrocardiogram*) dan EEG (*Electroencephalogram*). Selain itu ada juga metode non-intrusif yaitu dengan melakukan ekstraksi pada fitur wajah dengan menggunakan algoritma tertentu.

Salah satu pendekatan untuk mendeteksi kantuk adalah dengan cara intrusif, dimana fitur HRV (heart rate variability/variabilitas detak jantung) diekstraksi untuk merepresentasikan aktivitas sistem saraf otonom. Dalam penelitian yang dilakukan oleh Vicente, Laguna, Bartra, dan Bailón (2016), mereka mengamati 17 pria dan 13 wanita yang berpartisipasi dalam simulasi mengemudi. Nilai HRV diperoleh melalui penggunaan alat EEG (Elektroensefalogram - alat untuk mengukur aktivitas listrik otak) dan EKG (Elektrokardiogram - alat untuk mengukur aktivitas listrik jantung). Hasil penelitian menunjukkan bahwa detak jantung cenderung melemah seiring meningkatnya rasa kantuk (Vicente et al., 2016).

Pendekatan lain yang digunakan untuk mendeteksi kantuk adalah dengan cara non-intrusif, yaitu dengan melakukan ekstraksi pada fitur wajah dengan menggunakan citra wajah. Terdapat beberapa metode yang digunakan untuk menentukan area wajah diantaranya dengan menggunakan HOG, Linear SVM, haar cascade, dan segmentasi warna. Pada penelitian ini menggunakan metode Facial Landmark dengan versi 68-titik untuk mendekripsi area mata pada wajah. Dalam penelitian yang dilakukan Victorio (2022), penulis membuat model deep learning untuk mendekripsi kantuk menggunakan data uji UTA-RLDD. Untuk mengetahui apakah pengemudi dalam kondisi mengantuk adalah mengetahui pola bukaan mulut peserta ketika normal dan mengantuk. Hasilnya didapatkan model yang dapat mendekripsi rasa kantuk dengan akurasi

0.87 dan loss sebesar 0.65 (Valendion, 2022). Terdapat penelitian pengklasifikasikan tingkat kantuk seseorang yang dilakukan oleh Annida (2022), penulis membuat model deep learning untuk mengklasifikasikan kantuk menggunakan data uji DROZY. Untuk pengklasifikasian kantuk penulis menggunakan pola bukaan mata sebagai parameternya. Hasilnya didapatkan model yang dapat mengklasifikasikan kantuk dengan akurasi sebesar 0.976 dan loss sebesar 0.32 (Farodisa, 2022).

*Recurrent neural networks* (RNN) merupakan salah satu jenis dari algoritma *Deep Learning* yang banyak digunakan untuk tugas dengan input berurutan, seperti ucapan dan bahasa. Kinerja RNN lebih baik untuk data input berurutan karena mempunyai memori yang bisa dipergunakan untuk menyimpan state (LeCun et al., 2015). Pada tahun 2018, Shuali Li et al. mengatakan *Independently Recurrent Neural Network* (IndRNN) yakni variasi *deep learning* RNN. Bersumberkan pada penelitian Shuali Li et al. (2018), IndRNN ini 10 kali lebih cepat serta sedikit lebih akurat dari implementasi *deep learning Long Short-Term Memory* (LSTM) yang selalu dipergunakan (Li et al., 2018). Selain menggunakan *Deep Learning* model klasifikasi juga bisa dilakukan menggunakan *machine learning* salah satunya adalah SVM (*Support Vector Machine*). SVM menciptakan sebuah hiperplane atau sebuah garis pemisah yang optimal dalam ruang fitur untuk memisahkan data menjadi kelas-kelas yang berbeda. (Roy & Chakraborty, 2023)

Dengan akurasi pendekripsi kantuk yang baik maka berbagai masalah yang timbul dari rasa kantuk dihindari. Sebuah model untuk menentukan tingkat kantuk seseorang diperlukan karena dapat digunakan sebagai bahan penelitian yang berkaitan.

## 1.2 Rumusan Masalah

Bersumberkan latar belakang pada sub-bab 1.1, penggunaan EKG dan EEG dirasa tidak tepat untuk penggunaan sehari-hari karena memerlukan bantuan ahli untuk menggunakan alat-alat tersebut dengan tepat. Metode non intrusif berbasis citra terbukti dapat mendekripsi kantuk tidaknya seseorang dengan menggunakan *machine learning* namun tidak banyak penelitian yang membuat model klasifikasi kantuk dan kebanyakan hanya mengklasifikasikan kantuk berdasarkan fitur mata saja. Oleh karena itu diperlukan model klasifikasi kantuk berbasis citra dimana sistem akan mendekripsi rasa kantuk dengan mengimplementasikan *machine learning* untuk mengklasifikasikan tingkat kantuk pengemudi berdasarkan fitur mata dan mulut.

## 1.3 Tujuan

Tujuan dari penelitian ini adalah membangun sistem yang dapat mengklasifikasikan tingkat kantuk seseorang dengan parameter pola kedipan mata serta pola bukaan mulut yang dapat mengklasifikasikan kantuk menjadi 3 kelas berbeda yaitu waspada, kewaspadaan rendah, mengantuk (Ghoddoosian et al., 2019). Model yang diajukan adalah model klasifikasi kantuk dengan menggunakan IndRNN.

## 1.4 Batasan Masalah

Batasan-batasan dari proyek penelitian ini yaitu:

1. Data penelitian yang digunakan pada penelitian ini adalah video dari *The University of Texas at Arlington Real-Life Drowsiness Dataset (UTA-RLDD)*.
2. Model akan menglasifikasikan kantuk berdasarkan pola kedipan mata serta pola bukaan

dalam 3 kelas kantuk yaitu kelas 1 (Waspada), kelas 2 (Kewaspadaan Rendah), dan kelas 3 (Mengantuk)(Ghoddoosian et al., 2019).

3. Model akan memproses video masukan dengan durasi tercepat 9 menit dan dengan jumlah *frame per second* sebesar 30.

## 1.5 Manfaat

Manfaat dari penelitian ini yaitu

1. Bagi Penulis

Sebagai sarana implementasi ilmu yang telah didapatkan pada masa perkuliahan khususnya mengenai *computer vision* dan *machine learning*, melatih penulis untuk bisa berpikir secara logis, sistematis, dan kritis dalam pemecahan masalah sehingga dapat memecahkan permasalahan di kehidupan nyata seperti klasifikasi kantuk pada pengemudi.

2. Bagi Institusi

Sebagai pedoman serta inspirasi bagi penelitian lain yang berfokus pada pemecahan masalah klasifikasi kantuk menggunakan *machine learning* khususnya dengan metode IndRNN.

3. Bagi Masyarakat

Harapannya solusi yang dibuat pada penelitian ini bisa dilaksanakan serta digunakan menjadi produk nyata yang manfaatnya bisa dirasakan langsung oleh masyarakat sehingga dapat mengurangi dampak negatif dari rasa kantuk itu sendiri seperti digunakan pada *smart car*, mesin industri dan yang lainnya yang mempengaruhi produktivitas individu.

*[Halaman ini sengaja dikosongkan]*

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **2.1 Hasil penelitian terdahulu**

Berdasarkan tinjauan literatur yang mencakup beberapa penelitian mengenai pengembangan sistem deteksi kantuk secara non intrusif menggunakan pengolahan citra atau *machine learning*, penelitian sebelumnya menjadi relevan dan penting untuk dikaji. Mengacu pada referensi penelitian terdahulu akan membantu mencegah plagiarisme atau duplikasi hasil penelitian. Hal ini bertujuan untuk memberikan kontribusi bagi penulis dan memastikan bahwa penelitian mengenai tema ini dapat terus berkembang. Berikut adalah beberapa ulasan tentang penelitian terdahulu yang pernah dilakukan.

##### **2.1.1 Pengembangan Sistem Deteksi Kantuk Menggunakan Pengklasifikasi *Random Forest* pada Sinyal Elektrokardiogram**

Pada penelitian ini Nuryani Nuryani, Khoirun Nisak, Artono Dwijo Sutomo mengembangkan sistem deteksi kantuk memakai sinyal elektrokardiogram (EKG) dan *Random Forest*. Sistem deteksi kantuk dilatih memakai suatu rekaman EKG dari database DROZY yang dilengkapi *Karolinska Sleepiness Scale*(KSS). Fitur masukan sistem diekstraksi bersumberkan pada metode ranah waktu serta ranah frekuensi (Nisak, Sutomo, et al., 2021).

##### **2.1.2 Deteksi Kantuk Berdasarkan Pola Bukaan Mulut Menggunakan Metode 1D CNN**

Pada penelitian ini Victorio Valendion melakukan riset tentang deteksi kantuk memakai landmark bagian mulut melalui memperhitungkan pola dari terbuka serta tertutup-nya mulut. Nilai dari rasio bukaan mulut dihitung selanjutnya semua data nya disimpan ke dalam suatu file csv yang kemudian akan dilakukan proses pelatihan untuk mesin supaya dapat mendekripsi kantuk dengan arsitektur 1D CNN (Valendion, 2022).

##### **2.1.3 Klasifikasi Skala Kantuk Karolinska Berdasarkan Nilai *Eye Aspect Ratio* Menggunakan Deep Learning**

Pada penelitian ini Annida Miftakhul Farodisa melakukan riset tentang deteksi kantuk menggunakan metode non-intrusif dengan memanfaatkan kamera (fitur wajah) dan *deep learning*. Penulis membuat model klasifikasi Skala Kantuk Karolinska (SKK) berdasarkan nilai *Eye Aspect Ratio*. Penelitian ini menggunakan video *near-infrared* NIR pada dataset DROZY yang berisi video subjek saat menjalani pengujian kewaspadaan psikomotorik yang mengisi kuesioner Skala Kantuk Karolinska (SKK) sebelum memulai pengujian. Dilakukan ekstraksi nilai EAR untuk setiap subjek pada setiap frame di setiap video sehingga membentuk pola EAR dan dilabeli dengan SKK yang dikelompokkan menjadi empat kelas (Farodisa, 2022).

#### **2.1.4 Deteksi Kantuk Pada Pengendara Roda Empat Melalui Citra Wajah Menggunakan Metode *Facial Landmark***

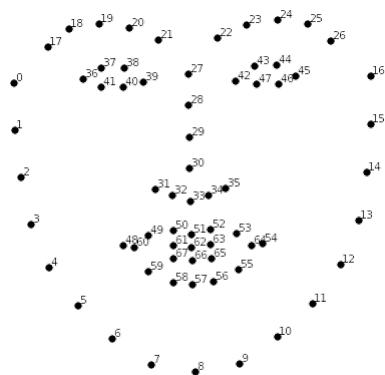
Pada penelitian ini Muhammad Akbar Maulana melakukan riset tentang deteksi kantuk pada pengendara roda empat melalui citra wajah memakai metode facial landmark. Membuat suatu sistem yang bisa memperingatkan pengemudi secara otomatis jika pengemudi memperlihatkan gejala mengantuk ketika mengendarai kendaraan akhirnya diharapkan bisa mengurangi resiko kecelakaan yang bisa terjadi kepada pengendara tersebut. Guna menetapkan pengendara mengantuk ataupun tidak sistem ini menggunakan algoritma *regression trees* dengan mengambil 2 parameter (*eye blink and Mouth detector*) (Maulana, 2022).

## 2.2 Library Dlib

Dlib merupakan library open source populer dalam bidang computer vision dan machine learning. Dlib menyediakan berbagai algoritma dan fungsi yang berguna untuk berbagai tugas seperti deteksi objek, pengenalan wajah, estimasi pose, deteksi landmark, pengolahan citra, dan lainnya. Dlib dapat berjalan di berbagai platform seperti Windows, macOS, Linux, dan Android (Ghoddoosian et al., 2019). Algoritma deteksi wajah Dlib menggunakan metode Histogram of Oriented Gradients (HOG) yang dikombinasikan dengan Support Vector Machines (SVM) untuk menghasilkan deteksi yang akurat. Dlib memiliki kemampuan untuk mendeteksi landmark pada wajah, yaitu titik-titik penting seperti mata, hidung, dan mulut. Deteksi landmark ini berguna dalam berbagai aplikasi seperti estimasi pose wajah, analisis ekspresi wajah, dan penerapan makeup virtual.

### 2.3 Facial Landmark

*Facial Landmark* merupakan ekstraksi titik-titik pada wajah yang tersebar menggunakan detektor wajah (Viola & Jones, 2004). Pendekripsi *facial landmark* diaplikasikan ke *input frames* untuk mendekripsi fitur wajah dengan menggunakan *library* dlib yang menggunakan *Histogram Oriented Gradients* (HOG) untuk pendekripsi wajah dan *Linear Support Vector Machine* (SVM) sebagai *classifier* dalam prosedur pengenalan landmark wajah (Ibrahim et al., 2021). Terdapat 68 titik yang merepresentasikan wajah, titik koordinat tersebut merepresentasikan fitur sesuai dengan gambar 2.1 dan dengan keterangan tiap koordinat pada tabel 2.1.



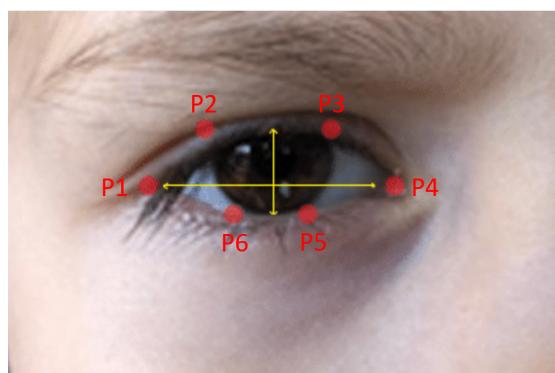
Gambar 2.1: Landmark wajah manusia (Ibrahim et al., 2021)

Tabel 2.1: Koordinat landmark wajah

No	Fitur	Koordinat Fitur
1	Dagu	0-16
2	Alis Kanan	17-21
3	Alis Kiri	22-26
4	Hidung	27-35
5	Mata Kanan	36-41
6	Mata Kiri	42-47
7	Mulut	48-60
8	Bibir	61-67

### 2.3.1 Eye Aspect Ratio (EAR)

EAR merupakan salah satu metode yang dapat digunakan untuk bisa menghitung jarak antara kelopak mata atas serta kelopak mata bawah bersumberkan titik geometri wajah pada mata. Metode EAR ini selalu dipergunakan untuk menghitung pola bukaan mata individual setiap menitnya. Perhitungan EAR ini dihitung bersumberkan pada koordinat mata kiri serta kanan yang ada dalam landmark wajah. EAR pada landmark wajah bisa diilustrasikan dengan gambar 2.2 :



Gambar 2.2: enam titik landmark pada mata (Rogalska et al., 2019)

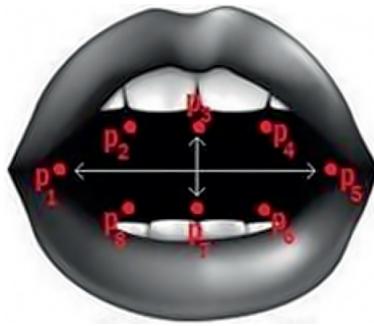
Nilai EAR didapati dari menghitung jarak antara kelopak mata yang didapatkan melalui mengambil titik tepat ditengah kelopak mata atas serta kelopak mata bawah. Rumus tersebut dapat dituliskan sesuai rumus 2.1:

$$EAR = \frac{\|P2 - P6\| + \|P3 - P5\|}{2 \|P1 - P4\|} \quad (2.1)$$

Ketika mata berkedip, dibutuhkan 100 - 400 ms untuk menutup. Perihal ini diperkuat oleh penelitian yang diselenggarakan oleh Caffier et al., yang menyatakan bahwasanya mata tertutup membutuhkan sekitar 200 ms. Sedangkan saat mengantuk, mata membutuhkan lebih dari 500 ms untuk menutup. Itu perkiraan mata tertutup dimulai saat mata mulai menutup sampai sepenuhnya tertutup (Wilkinson et al., 2013).

### 2.3.2 Mouth Aspect Ratio (MAR)

MAR ialah salah satu metode yang dapat dipergunakan untuk dapat menghitung jarak antara bibir atas dan bibir bawah bersumberkan titik geometri wajah pada mulut. Metode MAR ini selalu dipakai guna menghitung buaan mulut seseorang (Maulana, 2022). MAR pada landmark wajah bisa diilustrasikan dengan gambar 2.3 :



Gambar 2.3: Enam titik landmark pada mulut (Sri Mounika et al., 2022)

Perhitungan nilai MAR sama dengan perhitungan nilai EAR yaitu dengan menghitung jarak antara bibir yang diambil dengan mengambil titik tepat ditengah bibir atas dan bibir bawah. Rumus tersebut dapat dituliskan menjadi rumus 2.2:

$$MAR = \frac{\|P2 - P8\| + \|P3 - P7\| + \|P4 - P6\|}{3 \|P1 - P5\|} \quad (2.2)$$

### 2.4 Skala Kantuk Karolinska(SKK)

Skala Kantuk Karolinska menjadi alat yang paling banyak digunakan untuk perhitungan kewaspadaan dalam investigasi kronobiologis dan tidur. SKK ditemukan menjadi pengukur kantuk yang andal dan sensitif (Gillberg et al., 1994), dan itu divalidasi oleh beberapa pengukuran fisiologis yang berbeda (Kaida et al., 2006; Marzano et al., 2007; Filtness et al., 2021). Untuk mendeteksi transisi antara terjaga dan tidur dapat diukur menggunakan indeks fisiologis sederhana yang diambil dan ditentukan dari rekaman *electroencephalogram* (EEG). Nilai pada SKK berkisar antara 1 (amat sangat terjaga) hingga 9 (sangat mengantuk, berusaha lebih untuk terjaga) (Putilov & Donskaya, 2013). Hal tersebut dapat digambarkan menjadi seperti tabel ??:

Tabel 2.2: Skala Kantuk Karolinska (Putilov & Donskaya, 2013)

Skala	Tingkat Kantuk / Keterjagaan
1	Amat Sangat Terjaga
2	Sangat Terjaga
3	Terjaga
4	Sedikit Terjaga
5	Tidak Terjaga namun Tidak ada tanda Mengantuk
6	Sedikit Mengantuk
7	Mengantuk, sedikit usaha untuk Terjaga
8	Mengantuk, perlu usaha untuk Terjaga

Skala	Tingkat Kantuk / Keterjagaan
9	Sangat Mengantuk, berusaha lebih untuk Terjaga

## 2.5 Interpolasi Data

Interpolasi dilakukan untuk mengisi atau memperkirakan nilai yang hilang atau tidak tersedia dalam dataset dengan menggunakan teknik-teknik yang sesuai. Dalam beberapa kasus, dataset yang dikumpulkan mungkin tidak lengkap atau terdapat kehilangan data pada beberapa fitur. (Lepot et al., 2017). Beberapa teknik interpolasi data yang umum digunakan dalam machine learning meliputi:

1. Interpolasi Linear: Teknik ini melibatkan estimasi nilai yang hilang berdasarkan hubungan linear antara titik data yang diketahui. Interpolasi linear digunakan jika data memiliki hubungan linier yang jelas antara fitur-fitur yang terlibat.
2. Interpolasi Nearest Neighbor: Metode ini mengisi nilai yang hilang dengan menggunakan nilai terdekat dari data terdekat dalam dataset. data terdekat dapat ditentukan berdasarkan jarak Euclidean atau metode lainnya.
3. Interpolasi Polinomial: Metode ini melibatkan pendekatan fungsi polinomial untuk mengisi nilai yang hilang. Polinom dapat disesuaikan dengan data yang tersedia untuk memperkirakan nilai yang hilang.
4. Interpolasi Kriging: Teknik ini merupakan metode stokastik yang menggunakan model spasial untuk memprediksi nilai yang hilang berdasarkan korelasi spasial antara data yang diketahui.
5. Interpolasi Regresi: Pendekatan ini melibatkan penggunaan model regresi untuk memperkirakan nilai yang hilang berdasarkan hubungan antara atribut-atribut yang tersedia.

## 2.6 Normalisasi

Normalisasi adalah proses mengubah skala nilai dari fitur-fitur dalam dataset agar memiliki distribusi data yang seragam atau standarisasi. Tujuan normalisasi adalah untuk memastikan bahwa fitur-fitur dengan skala yang berbeda memiliki pengaruh yang sebanding terhadap proses pembelajaran algoritma (Raju et al., 2020). Beberapa metode normalisasi umum digunakan adalah sebagai berikut:

1. Min-Max Scaling: Metode ini mentransformasikan setiap nilai fitur menjadi rentang yang ditentukan, biasanya antara 0 dan 1. Rumus umum untuk melakukan normalisasi Min-Max Scaling adalah sebagai berikut:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (2.3)$$

Pada rumus 2.3 X merupakan nilai asli, X' merupakan nilai yang telah dinormalisasi, X<sub>min</sub> adalah nilai minimum yang ada pada dataset, dan X<sub>max</sub> adalah nilai maksimum yang ada pada dataset.

2. Z-Score Scaling: Metode ini mengubah nilai fitur menjadi distribusi standar dengan mean (rata-rata) 0 dan standard deviation (deviasi standar) 1. Rumus umum untuk melakukan

normalisasi Z-Score Scaling adalah sebagai berikut:

$$X' = \frac{X - \text{mean}}{\text{stdDev}} \quad (2.4)$$

Pada rumus 2.4 X merupakan nilai asli, X' merupakan nilai yang telah dinormalisasi, mean merupakan nilai rata-rata fitur yang ada pada dataset, dan stdDev merupakan nilai standar deviasi dari fitur yang ada pada dataset.

## 2.7 Augmentasi Data

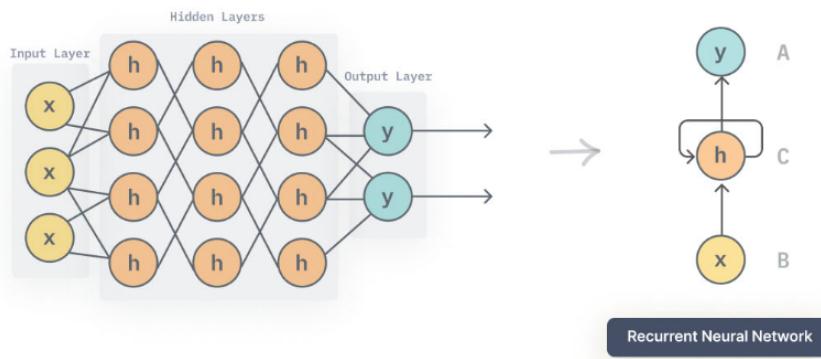
Augmentasi data merupakan teknik yang digunakan untuk memperbanyak jumlah data dengan cara memodifikasi data yang telah ada sehingga dapat meningkatkan variasi dalam dataset pelatihan tanpa harus mengumpulkan lebih banyak data aktual. Jumlah data yang sedikit menjadi alasan dilakukannya teknik augmentasi data. Data baru yang diperoleh dari proses augmentasi data dapat digunakan sebagai masukan bagi model *machine learning*. Augmentasi data dapat meningkatkan performa model dapat dan digunakan untuk mencegah overfitting serta meningkatkan kemampuan model untuk menggeneralisasi dengan lebih baik. Teknik umum dalam melakukan augmentasi data diantaranya pemutaran dan pemotongan, pergeseran, perubahan skala, pencerahan dan Kontras, pembalikan, perubahan warna. (Sanjaya & Ayub, 2020)

## 2.8 Deep Learning

*Deep Learning* memungkinkan model komputasi yang memuat atas sejumlah lapisan pemrosesan guna memahami representasi data melalui sejumlah tingkatan abstraksi. Metode-metode ini sudah meningkatkan *state-of-the-art* dalam pengenalan suara, pengenalan objek visual, deteksi objek serta banyak domain lainnya secara dramatis. *Deep Learning* mendapati struktur rumit dalam kumpulan data besar yang memakai algoritma backpropagation guna memperhatikan bagaimana mesin perlu merubah parameter internalnya yang dipakai guna memperhitungkan representasi di setiap lapisan dari representasi di lapisan sebelumnya. *Deep Convolutional Network* mendapatkan terobosan dalam pemrosesan gambar, video, ucapan, serta audio, sementara *Recurrent Neural Network* menyoroti data berurutan berupa teks serta ucapan (Le-Cun et al., 2015).

## 2.9 Recurrent Neural Network (RNN)

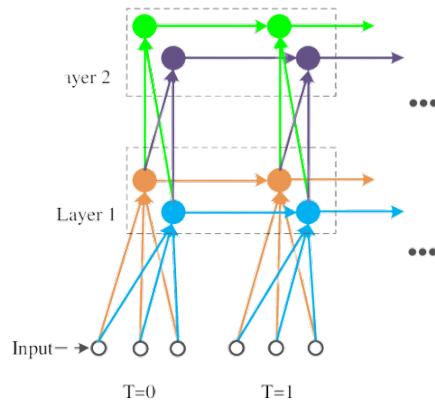
Untuk tugas yang melibatkan input berurutan, seperti ucapan dan bahasa, seringkali lebih baik untuk menggunakan RNN. RNN mempunyai memori yang bisa dipergunakan untuk menyimpan state, memori ini memungkinkan RNN untuk memiliki pemahaman tentang konteks sebelumnya saat memproses langkah berikutnya. RNN memproses urutan input satu elemen pada setiap waktu, mempertahankan di unit tersembunyi mereka sebuah *state vektor* yang secara implisit berisi informasi tentang sejarah semua elemen masa lalu dari urutannya yang bisa diilustrasikan dengan gambar 2.4.



Gambar 2.4: Ilustrasi RNN (LeCun et al., 2015)

RNN menggunakan bobot yang bersama-sama digunakan di setiap langkah, dengan tujuan menghasilkan pembaruan yang konsisten pada setiap langkah. Bobot ini mencerminkan hubungan antara langkah sebelumnya dan langkah berikutnya. Ketika kita mempertimbangkan output dari unit tersembunyi di langkah-langkah waktu diskrit yang berbeda seolah-olah mereka adalah output dari yang berbeda neuron dalam *deep multilayer network*. Backpropagation digunakan untuk melatih RNN dengan menghitung gradien kesalahan melalui setiap langkah waktu dan memperbarui bobot-bobot berdasarkan gradien tersebut. Hal ini memungkinkan RNN untuk belajar dan menyesuaikan diri dengan pola dalam data urutan.(LeCun et al., 2015).

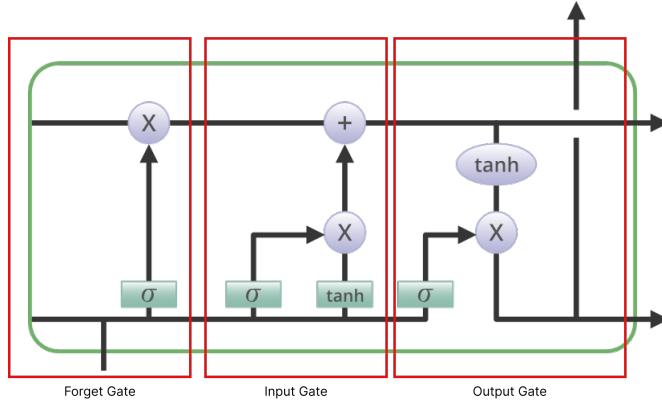
## 2.10 Independent RNN (IndRNN)



Gambar 2.5: Ilustrasi IndRNN (Li et al., 2018)

IndRNN merupakan salah satu variasi *deep learning* hasil pengembangan RNN yang bisa mengetahui pola temporal. IndRNN ini diperlihatkan oleh Shuali Li et al. (2019). IndRNN memperkenalkan gagasan independensi di antara unit-unit waktu dalam jaringan rekuren, sehingga setiap unit rekuren dapat beroperasi secara mandiri tanpa ketergantungan langsung pada unit-unit sebelumnya atau sesudahnya. sedangkan pada RNN, setiap neuron terkait satu dengan lainnya dalam satu layer sesuai gambar 2.5. Pemrosesan neuron yang independen ini memungkinkan IndRNN untuk memproses urutan panjang dengan menggunakan parameter yang sama pada setiap unit rekuren. Keunggulan IndRNN terletak pada kemampuannya untuk memproses urutan waktu yang panjang dengan menggunakan parameter yang lebih efisien dan menghindari *vanishing gradient problem* yang sering terjadi pada RNN tradisional.

## 2.11 Long Short Terms Memory (LSTM)



Gambar 2.6: Struktur LSTM

LSTM merupakan pengembangan atau salah satu jenis dari algoritma RNN (*Recurrent Neural Network*) yaitu dengan menambahkan memory cell yang dapat menyimpan informasi untuk jangka waktu yang lama (Manaswi & Manaswi, 2018). LSTM diajukan sebagai solusi untuk mengatasi masalah *vanishing gradient* yang terjadi saat memproses data berurutan yang panjang saat menggunakan RNN. LSTM memiliki kemampuan untuk mengingat informasi yang telah disimpan dalam jangka waktu panjang dan menghapus informasi yang tidak relevan sehingga memungkinkannya untuk lebih efisien memproses, memprediksi, dan mengklasifikasikan data berdasarkan urutan waktu tertentu. LSTM juga melibatkan proses backpropagation untuk melatih dan memperbarui bobot-bobotnya. Gradien dihitung mundur melalui waktu untuk memperbarui bobot-bobot secara efisien.

Sistem LSTM terdiri dari empat gerbang, yaitu masukan (*input gate*), gerbang lupa (*forget gate*), gerbang dan gerbang keluaran (*output gate*) (Chung & Shin, 2018). Setiap gerbang memiliki peran dan tugasnya sendiri dalam mengumpulkan, mengklasifikasi, dan memproses data. Selain memiliki empat gerbang tersebut, LSTM juga dilengkapi dengan sel memori internal. Berikut penjelasan rinci mengenai komponen-komponen pada LSTM:

### 1. Gerbang Input (*Input Gate*)

Gerbang input mengontrol seberapa banyak informasi baru harus dimasukkan ke dalam sel memori. Ini melibatkan penggunaan fungsi aktivasi sigmoid untuk menghasilkan vektor bobot yang memperkirakan tingkat pentingnya informasi baru.

### 2. Gerbang Lupa (*Forget Gate*)

Gerbang lupa menentukan seberapa banyak informasi lama dalam sel memori harus dilupakan. Ini juga menggunakan fungsi aktivasi sigmoid untuk menghasilkan vektor bobot yang menentukan sejauh mana informasi lama harus dihapus atau dipertahankan.

### 3. Sel Memori

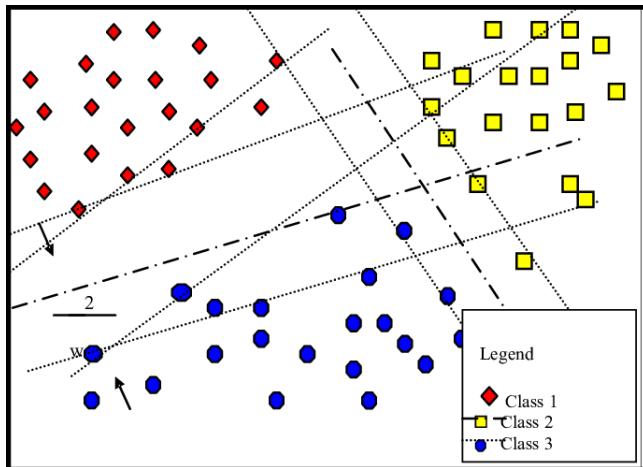
Sel memori merupakan komponen sentral dalam LSTM. Ini berfungsi sebagai "memori" jangka panjang yang dapat menyimpan dan mengingat informasi dalam urutan data. Sel memori dikontrol oleh gerbang input dan gerbang lupa untuk memperbarui dan mempertahankan informasi yang relevan.

### 4. Gerbang Keluaran (*Output Gate*)

Gerbang keluaran mengatur sejauh mana informasi dalam sel memori harus diungkapkan ke luar. Fungsi aktivasi sigmoid digunakan untuk menghasilkan vektor bobot yang mengontrol jumlah informasi yang dilewatkan.

## 2.12 Support Vector Machine (SVM)

SVM (*Support Vector Machine*) adalah sebuah algoritma pembelajaran mesin yang digunakan untuk tugas klasifikasi dan regresi. SVM menciptakan sebuah hiperplane atau sebuah garis pemisah yang optimal dalam ruang fitur untuk memisahkan data menjadi kelas yang berbeda. (Roy & Chakraborty, 2023)



Gambar 2.7: SVM multiklas

### 1. Klasifikasi Linear:

SVM memiliki kemampuan untuk melakukan klasifikasi linear, di mana algoritma ini mencari hiperplane yang secara optimal memisahkan dua kelas data. Hiperplane tersebut dapat berupa garis atau bidang yang berfungsi sebagai pemisah. SVM berupaya menemukan hiperplane dengan margin terbesar di antara dua kelas, yaitu jarak terpendek antara hiperplane dan titik-titik data terdekat dari masing-masing kelas.

### 2. Kernel:

SVM juga memiliki kemampuan untuk memodelkan hubungan non-linear antara fitur-fitur dengan menggunakan kernel. Kernel merupakan suatu fungsi yang mengubah data ke dalam ruang fitur yang memiliki dimensi lebih tinggi. Dalam ruang fitur yang lebih tinggi tersebut, SVM mencari hiperplane linier yang mampu memisahkan data secara optimal. Beberapa jenis kernel yang sering digunakan meliputi kernel linier, kernel polinomial, dan kernel radial basis function (RBF).

### 3. Support Vectors:

Support vectors adalah titik-titik data yang terletak pada atau dekat hiperplane pembatas. SVM menggunakan support vectors ini untuk menentukan hiperplane yang optimal. Hanya support vectors yang mempengaruhi posisi dan bentuk hiperplane, sementara titik-titik data lainnya tidak berperan dalam pembentukan hiperplane.

### 4. Regularisasi dan Penalti:

SVM menerapkan konsep regularisasi untuk mencegah overfitting. Parameter  $C$  dalam SVM mengatur keseimbangan antara mencapai margin maksimum dan mengurangi jumlah pelanggaran batas pemisah (misclassification). Semakin besar nilai  $C$ , semakin besar penalti yang diberikan untuk pelanggaran batas pemisah. Dengan demikian, nilai  $C$  yang lebih tinggi cenderung menghasilkan model yang lebih ketat dengan margin yang lebih sempit, sementara nilai  $C$  yang lebih rendah dapat memungkinkan adanya pelanggaran batas pemisah yang lebih besar.

5. Regresi:

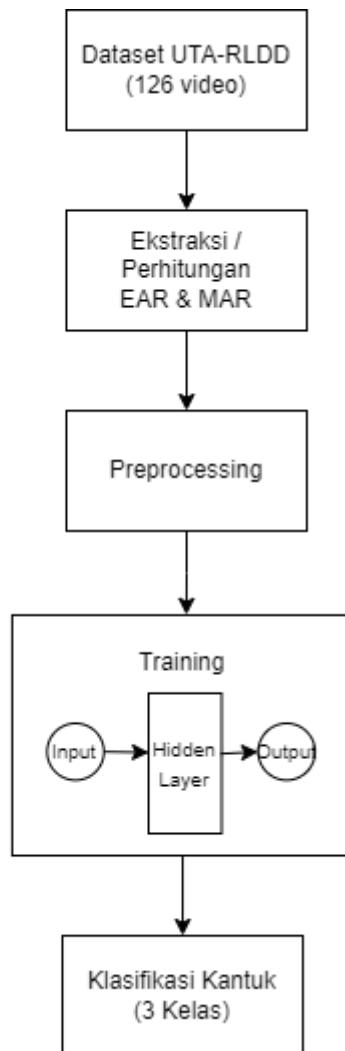
Selain untuk klasifikasi, SVM juga dapat diterapkan dalam tugas regresi. Dalam SVM regresi, tujuannya adalah untuk membangun sebuah model yang dapat mendekati fungsi target dengan cara meminimalkan kesalahan antara output model dan target sebanyak mungkin. SVM regresi berusaha untuk menciptakan sebuah hiperplane yang memiliki margin terbesar di sekitar titik-titik data yang ada.

## BAB III

# METODOLOGI

Pada bab metodologi akan dijelaskan mengenai bagaimana cara sistem ini dibuat. Tujuan dari sistem ini adalah membuat model klasifikasi tingkat kantuk seseorang dengan parameter EAR dan MAR menggunakan IndRNN dengan *output* 3 kelas berbeda.

### 3.1 Metode yang digunakan



Gambar 3.1: Diagram alur model

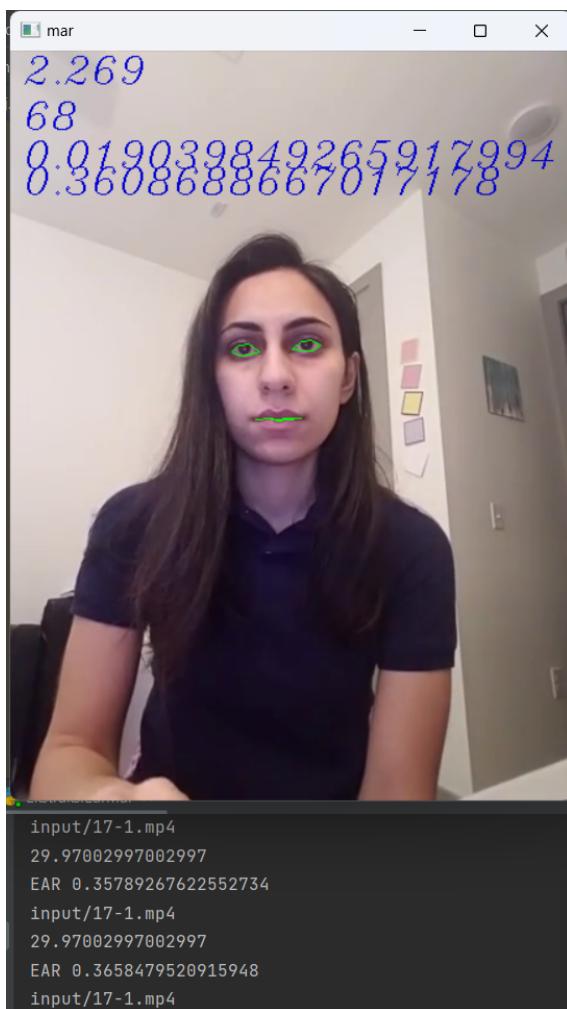
Pada diagram yang tertera di Gambar 3.1. Didapatkan proses sebagai berikut:

### 3.1.1 Pengumpulan Data

Dataset UTA-RLDD digunakan sebagai sumber data input. Setiap video dalam kumpulan data ini memiliki perkiraan durasi 10 menit. Pengumpulan data dilakukan secara diseleksi secara manual. Dataset UTA-RLDD memiliki subjek video yang sangat beragam mulai dari jenis kelamin, etnis, kegiatan yang dilakukan, ukuran video, keberadaan rambut wajah, dan pemakaian kacamata. Terdapat beberapa video yang tidak bisa terbuka sehingga perlu dilakukan penyaringan data. Adapun kriteria dari data yang dipilih adalah:

1. Video memiliki fps mendekati 30fps.
2. Video memiliki jumlah frame diatas 16200.
3. Video memiliki pencahayaan yang baik.

### 3.1.2 Ekstraksi Fitur



Gambar 3.2: Proses ekstraksi nilai EAR dan MAR

Ekstraksi fitur pada tiap video yang telah terpilih dilakukan secara lokal dengan bantuan IDE PyCharm edisi 2021.3.3. Library Dlib menggunakan pendekatan berbasis fitur untuk mendeteksi landmark wajah. Algoritma pada library Dlib menggunakan metode Histogram of Oriented Gradients (HOG) yang dikombinasikan dengan *Support Vector Machines* (SVM) untuk mengenali pola wajah. Deteksi wajah ini akan menghasilkan kotak pembatas (*bounding*

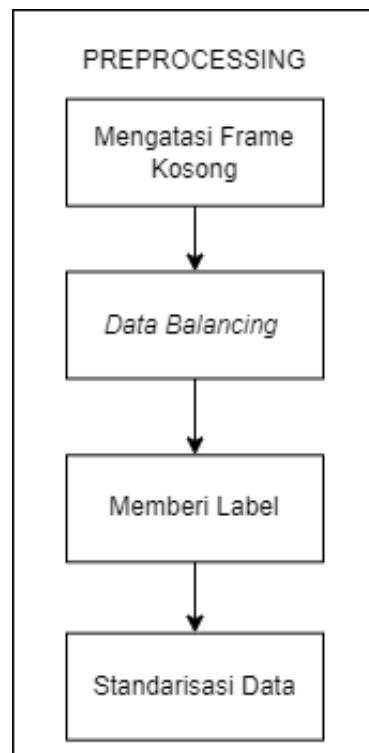
*box*) yang mengelilingi wajah dalam video. Setelah mendeteksi wajah, langkah selanjutnya adalah menggunakan prediktor landmark Dlib untuk mengidentifikasi titik-titik landmark pada wajah.

Prediktor landmark menggunakan metode regresi untuk memprediksi koordinat titik landmark pada wajah berdasarkan fitur-fitur yang terlihat dalam gambar. Prediktor ini mengambil gambar wajah yang terdeteksi sebagai input dan menghasilkan koordinat landmark yang sesuai. Koordinat 68 titik landmark ini mencakup fitur-fitur seperti mata, hidung, mulut, dan sebagainya (Kazemi & Sullivan, 2014). Setiap titik landmark direpresentasikan oleh sepasang koordinat (x, y) yang menunjukkan posisinya dalam gambar seperti gambar 3.2.

Fitur-fitur yang diekstraksi pada tiap video yaitu nilai dari EAR dan MAR. Rumus 2.1 digunakan untuk mendapatkan nilai EAR serta rumus 2.2 untuk mendapatkan nilai MAR. Rumus-rumus tersebut kemudian dimasukkan ke dalam program sehingga menghasilkan data berupa csv yang berisi nilai EAR dan MAR dari tiap frame dalam setiap video. Nilai dari EAR dan MAR, akan didapatkan pada tiap frame berdasarkan waktu.

### 3.1.3 Preprocessing Data

Preprocessing data dilakukan menggunakan GoogleColab. Preprocessing data dilakukan untuk meningkatkan kualitas data, mengurangi noise, dan membuat data siap digunakan untuk pelatihan model. Tahapan preprocessing data yang telah dilakukan dalam penelitian ini adalah mengatasi data kosong, penyetaraan jumlah data dan pemberian label, melakukan standarisasi data, serta melakukan kostumisasi pada data sehingga cocok digunakan untuk IndRNN. Tahapan-tahapan tersebut dapat digambarkan menjadi gambar 3.3:



Gambar 3.3: Diagram Preprocessing Data

## 1. Interpolasi Linear

Terdapat beberapa frame dari video yang landmark wajahnya tidak terdeteksi sehingga menyebabkan koordinat landmark wajah bernilai NaN. Interpolasi linear diperlukan untuk memperkirakan nilai di antara dua titik data yang diketahui. Interpolasi linear mengacu pada estimasi nilai di antara dua titik data yang diberikan berdasarkan garis lurus yang menghubungkan kedua titik tersebut. Konsep dasar interpolasi linear adalah menggunakan persamaan garis untuk memperkirakan nilai di antara dua titik data. Persamaan garis linear dinyatakan sebagai persamaan 3.1:

$$y = mx + c \quad (3.1)$$

Setelah nilai koordinat tiap frame tidak ada yang bernilai NaN maka nilai EAR dan MAR dapat kembali dihitung menggunakan hasil interpolasi koordinat titik-titik yang ada pada mata dan mulut dengan memasukkannya pada rumus 2.1 dan rumus 2.2.

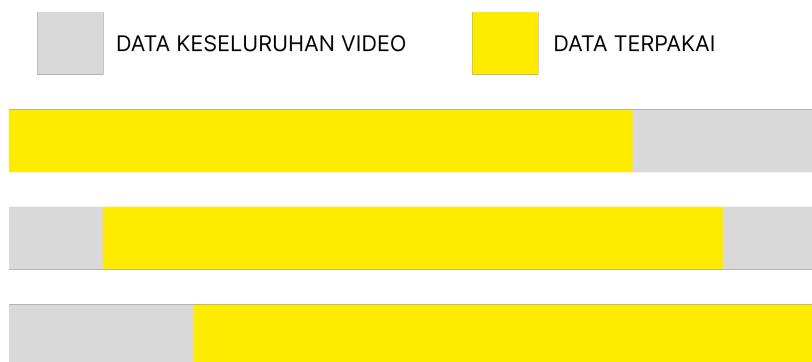
## 2. *Data Balancing* dan *Labeling*

Jumlah frame yang ada pada tiap video tidaklah sama oleh karena itu diperlukan pemotongan jumlah frame pada tiap video sehingga jumlah framenya dapat seragam yaitu 16200 frame tiap video. Ketika jumlah frame tiap video sudah seragam maka diberikanlah label yang sama pada tiap frame dalam satu video. Data-data tersebut kemudian dijadikan dalam satu csv sehingga lebih mudah untuk melakukan tahap preprocessing data selanjutnya.

## 3. Normalisasi Standard Scaler

Metode normalisasi standard scaler umum digunakan dalam preprocessing data. Tujuannya adalah untuk mentransformasikan data sehingga memiliki mean (rerata) nol dan deviasi standar satu. Normalisasi standard scaler dapat membantu meningkatkan performa model serta menghilangkan perbedaan skala antar fitur dalam dataset. Dengan demikian, semua fitur akan memiliki skala yang serupa, yang memudahkan perbandingan dan interpretasi data. Normalisasi data dengan fungsi Standard Scaler dapat diakses melalui library sklearn. Cara kerja dari normalisasi ini adalah mengganti nilai tersebut dengan nilai yang dihasilkan melalui rumus 2.4.

## 4. Sliding Window



Gambar 3.4: Ilustrasi pemotongan data

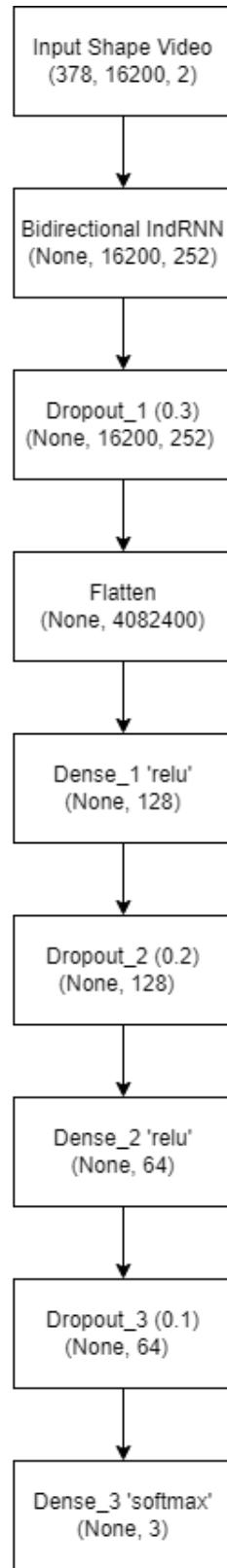
Sebelum data dimasukkan ke sliding window, data terlebih dahulu dipisahkan menjadi data training dan data validasi atau testing. Sliding window digunakan sebagai teknik pemrosesan data untuk mempersiapkan data masukan yang lebih sesuai dengan kebutuhan model. Metode sliding window digunakan karena masukan feature bersifat kontinu

dan dalam satu feature terdapat banyak nilai yang tersimpan di dalam array. Dilakukan augmentasi pada data training dengan melakukan pemotongan data pada titik acu yang berbeda. Jumlah data bervariasi dalam tiap video namun data yang diambil sebagai masukan model berjumlah 16200. Oleh karena itu dilakukan pemotongan dengan titik acu yang berbeda untuk menghasilkan data augmentasi. Pemotongan dilakukan dengan variasi dari awal data, dari tengah data, dan dari akhir data yang dapat diilustrasikan seperti gambar 3.4.

### **3.1.4 *Training Data***

Pada proses klasifikasi yang menggunakan IndRNN nilai EAR dan MAR digunakan sebagai feature untuk proses training. Untuk output-nya sendiri akan terdiri dari 3 kelas yang mengacu pada skala kantuk karolinska. 3 kelas tersebut terdiri dari: Kelas pertama (Waspada) yaitu terdiri dari label kelas KSS 1 sampai 5 ( Amat Sangat Terjaga, Sangat Terjaga, Terjaga, Sedikit terjaga, Tidak Terjaga namun Tidak ada Tanda Mengantuk), Kelas kedua (Kewaspadaan Rendah) yaitu terdiri dari label kelas KSS 6 dan 7 ( Sedikit Mengantuk dan Mengantuk, perlu sedikit usaha untuk terjaga), Kelas ketiga (Mengantuk) yaitu terdiri dari label kelas KSS 8 dan 9 (Mengantuk, perlu usaha untuk terjaga dan Sangat mengantuk, perlu usaha lebih untuk tetap terjaga). Model klasifikasi dibuat menggunakan IndRNN sebagai klasifikatornya.

Berikut adalah gambaran arsitektur IndRNN yang digunakan:



Gambar 3.5: Arsitektur Model Klasifikasi IndRNN

1. Input: IndRNN menerima input berupa urutan data, seperti urutan waktu dalam bentuk deret waktu atau urutan spasial dalam citra. Setiap elemen dalam urutan dianggap sebagai input pada satu waktu tertentu. IndRNN memiliki input yang berbentuk 3 dimensi jika

pada penelitian yang dilakukan, bentuk input modelnya adalah (353, 16200, 2).

2. Hidden Units: Setiap unit rekuren dalam IndRNN memiliki *hidden state* atau *output* yang dihasilkan dari pemrosesan input dan bobotnya. Hidden state ini merepresentasikan informasi yang dihasilkan oleh unit rekuren pada waktu tertentu. Model klasifikasi pada penelitian ini memiliki 126 hidden units.
3. Weight Matrix: Setiap unit rekuren dalam IndRNN menggunakan *weight matrix* (matriks bobot) yang berkorelasi dengan dirinya sendiri. Matriks bobot ini digunakan dalam operasi perkalian untuk menghitung *output* unit rekuren pada waktu tertentu.
4. Non-linear Activation: Setelah melakukan operasi perkalian dengan input dan bobotnya, output dari setiap unit rekuren umumnya melewati fungsi aktivasi non-linear, seperti fungsi ReLU, untuk memperkenalkan sifat non-linearitas dalam jaringan.
5. *Output*: IndRNN dapat menghasilkan output pada setiap waktu tertentu berdasarkan hidden state dari unit-unit rekuren. *Output* ini dapat digunakan untuk klasifikasi kantuk yang menghasilkan 3 jenis kelas berbeda.

Model klasifikasi yang dibuat dengan arsitektur SVM, menggunakan parameter kernel RBF. Kernel RBF (*Radial Basis Function*) atau juga dikenal sebagai Gaussian kernel adalah salah satu jenis kernel yang memetakan data ke dalam ruang fitur yang memiliki dimensi tak terhingga. Kernel RBF memberikan bobot pada setiap titik data berdasarkan jaraknya dari pusat. Semakin dekat titik data dengan pusat, semakin besar bobotnya. Dalam kernel RBF, bobot jarak dihitung dengan menggunakan fungsi Gaussian, yang menurun secara eksponensial dengan jarak. Kernel RBF digunakan untuk memodelkan hubungan non-linear antara fitur-fitur dalam data. Dengan melakukan pemetaan data ke dalam ruang fitur yang memiliki dimensi tak terhingga, kernel RBF memungkinkan SVM untuk menemukan hiperplane non-linear yang dapat memisahkan kelas-kelas data dengan lebih baik.

Berikut merupakan gambaran arsitektur LSTM yang digunakan sebagai model klasifikasi kantuk:

```

Model: "sequential"

-----  

Layer (type)          Output Shape       Param #  

-----  

bidirectional (Bidirectiona (None, 16200, 252)      130032  

l)  

dropout (Dropout)      (None, 16200, 252)      0  

Flatten (Flatten)      (None, 4082400)        0  

Dense_3 (Dense)        (None, 64)            261273664  

dropout_1 (Dropout)     (None, 64)            0  

Dense_4 (Dense)        (None, 32)            2080  

dropout_2 (Dropout)     (None, 32)            0  

dense (Dense)          (None, 3)             99  

-----  

Total params: 261,405,875  

Trainable params: 261,405,875  

Non-trainable params: 0
-----
```

Gambar 3.6: Arsitektur Model Klasifikasi LSTM

Berikut merupakan tabel hyperparameter yang digunakan untuk membangun model klasifikasi pada penelitian ini:

Tabel 3.1: Pengaturan Hyperparameter Model Klasifikasi

Hyperparameter	Nilai
Arsitektur Model (IndRNN, LSTM, SVM)	126 unit
Regularisasi L2	1e-07
Dropout_1	0.3
Dropout_2	0.2
Dropout_3	0.1
activation	ReLU
output activation	softmax
solver	adam
initial learning rate	0.000025

### 1. Regularisasi L2

Regularisasi L2 adalah salah satu metode regularisasi dalam pembelajaran mesin. Tujuan dari digunakannya regularisasi L2 adalah untuk mengendalikan kompleksitas model dan mencegah overfitting. Regulasi L2 biasa digunakan pada data yang kompleks yang menyebabkan kondisi model terlalu "menghafal" data pelatihan dan tidak mampu menggeneralisasi dengan baik pada data yang belum pernah dilihat sebelumnya. Regularisasi L2 membantu mencegah overfitting dengan mengurangi bobot yang besar, sehingga membatasi kapasitas model untuk "menghafal" data pelatihan dan memaksa model untuk

fokus pada pola yang lebih umum dan general. Regulasi L2 juga dapat membuat model lebih stabil dan konsisten dengan mengendalikan varians serta mengendalikan kemampuan model dengan memberikan penalti pada bobot yang lebih besar, sehingga mendorong model untuk menggunakan representasi yang lebih sederhana dan parsial dari data.

### 2. Fungsi Aktivasi Non-Linear

Pada model klasifikasi kantuk dengan IndRNN, ReLU digunakan untuk memperkenalkan sifat non-linearitas dalam jaringan. Alasan digunakannya ReLU pada model klasifikasi karena implementasinya sederhana, perhitungan ReLU dapat dijalankan secara efisien di perangkat keras seperti GPU. ReLU dapat mengurangi masalah gradien yang cenderung menghilang (*vanishing gradient*) saat mengalami backpropagation, terutama pada lapisan-lapisan terdalam. ReLU memungkinkan pelatihan model yang lebih cepat dan efisien. Fungsi aktivasi softmax digunakan pada lapisan dense terakhir dengan tujuan menghasilkan distribusi probabilitas yang menunjukkan kemungkinan setiap kelas sebagai output dari model.

### 3. Optimisasi

Optimisasi Adam (Adaptive Moment Estimation) adalah algoritma optimasi yang menggabungkan konsep dari optimizer RMSProp dan Momentum untuk menghasilkan tingkat konvergensi yang cepat dan efisien. Optimisasi Adam menghitung momen pertama dan kedua dari gradien. Momen pertama digunakan untuk mengestimasi kecepatan perubahan bobot, sedangkan momen kedua digunakan untuk mengestimasi kecepatan perubahan kecepatan perubahan bobot. Hal ini membantu algoritma untuk menyesuaikan langkah optimisasi berdasarkan gradien yang terkini dan mempertimbangkan sejarah gradien sebelumnya. Dengan melakukan penyesuaian learning rate berdasarkan estimasi momen, optimisasi Adam mampu menyesuaikan langkah optimisasi dengan lebih baik dan mengurangi dampak dari learning rate yang tidak tepat. Optimisasi Adam melibatkan regularisasi untuk mencegah overfitting pada model. Regularisasi dilakukan melalui metode L2 regularization. Optimizer Adam digunakan untuk mempercepat dan meningkatkan proses pelatihan model IndRNN, sehingga memungkinkan model untuk belajar pola yang lebih kompleks dan akurat dari data sequence yang ada.

## 3.1.5 Evaluasi Model

Evaluasi model dilakukan untuk memahami dan mengukur performa model dalam melakukan prediksi pada dataset. Beberapa metrik evaluasi yang umum digunakan adalah Confusion Matrix, Akurasi, Presisi, Recall, dan F1-Score. Semakin tinggi nilai precision, recall, dan f1 score maka semakin baik performa model. Nilai loss yang mendekati 0 menandakan hasil probabilitas prediksi mendekati nilai yang benar dan juga sebaliknya. Nilai yang benar pada confusion matrix dapat diliat dari garis diagonalnya. Tahapan skenario evaluasi model meliputi beberapa poin berikut:

1. Pengujian Performa Model IndRNN dengan Confusion Matrix
2. Pengujian Performa Model IndRNN dengan Akurasi, Presisi, Recall, dan F1-Score
3. Pengujian Performa Model LSTM dengan Confusion Matrix
4. Pengujian Performa Model LSTM dengan Akurasi, Presisi, Recall, dan F1-Score
5. Pengujian Performa Model SVM dengan Confusion Matrix
6. Pengujian Performa Model SVM dengan Akurasi, Presisi, Recall, dan F1-Score

### 3.1.6 Pengujian Model

Pengujian model digunakan untuk mengetahui kinerja model yang telah dibuat. Pengujian model dilakukan dengan menggunakan video yang ada pada dataset DROZY. Video yang ada pada dataset diproses dahulu seperti video yang berasal dari dataset UTA-RLDD sehingga menghasilkan data yang sesuai. Video yang memiliki jumlah frame yang kurang dari 16200 tidak digunakan untuk melakukan pengujian model.

## 3.2 Data dan peralatan yang digunakan

Model klasifikasi kantuk dibangun menggunakan data dari dataset UTA-RLDD serta untuk menguji model klasifikasi digunakan dataset DROZY. Alat yang dibutuhkan untuk membangun model ini adalah laptop yang terhubung dengan internet sehingga dapat digunakan untuk mengakses GoogleColab dan PyCharm.

### 3.2.1 Dataset UTA-RLDD



Gambar 3.7: Sampel data UTA-RLDD dalam status waspada (baris pertama), waspada rendah (baris kedua), dan mengantuk (baris ketiga) (Ghoddoosian et al., 2019)

UTA-RLDD (*The University of Texas at Arlington Real-Life Drowsiness Dataset*) dibuat untuk tugas deteksi kantuk *multi-stages*. Dataset ini terdiri dari 60 subjek sehat (51 laki-laki dan 9 perempuan) dengan usia diatas 18 tahun. Panjang durasi jumlah video yang ada sekitar 30jam RGB dengan fps hingga 30. Subjek diminta untuk mengambil 3 buah video mereka dalam 3 kondisi kantuk yang berbeda seperti pada gambar 3.7 dengan durasi masing-masing sekitar 10 menit(Ghoddoosian et al., 2019). Ketiga kelas tersebut dijelaskan kepada para peserta sebagai berikut:

1) Waspada: Salah satu dari tiga kondisi pertama yang disorot dalam tabel KSS dengan nilai 1 hingga 3 dimana subyek dalam keadaan masih waspada dan benar-benar sadar sehingga mereka dapat mengemudi selama berjam-jam dengan mudah.

2) Kewaspadaan Rendah : Memiliki nilai 6 dan 7 dalam tabel KSS dimana ketika beberapa tanda kantuk muncul, atau kantuk hadir tetapi tidak diperlukan upaya untuk tetap waspada.

Subjek mungkin dapat mengemudi dalam kondisi ini tapi tidak dianjurkan.

3) Mengantuk : Keadaan ini berarti subjek harus secara aktif berusaha untuk tidak tertidur dengan nilai 8 dan 9 pada Tabel KSS.

### 3.2.2 Dataset DROZY



Gambar 3.8: Sampel data DROZY (Massoz et al., 2016)

Database DROZY berisi berbagai tipe data yang berhubungan dengan kantuk (sinyal, gambar, dan sebagainya). Sampel data yang ada pada database DROZY ditunjukkan oleh gambar 3.8. Database ini bertujuan untuk membantu para peneliti untuk melakukan eksperimen, mengembangkan, serta mengevaluasi sistem pada monitoring kantuk. Data yang ada di DROZY diambil dari 14 partisipan (3 pria, 11 wanita) yang melakukan simulasi berkendara selama 10 menit dalam kondisi kurang tidur yang disebabkan oleh bangun yang berkepanjangan. Untuk setiap subjek, dan untuk setiap PVT, database berisi data tersinkronisasi waktu dengan sem-purna berikut:

- 1) Nilai KSS
- 2) PVT data
- 3) Polysomnography signals
- 4) Kinect V2 sensor videos
- 5) Face Landmarks (Manual)
- 6) Face Landmarks (Otomatis)
- 7) Index Interpolasi

### 3.2.3 Peralatan

Berikut merupakan peralatan yang diperlukan untuk membangun model klasifikasi kantuk dengan menggunakan IndRNN:

1. Laptop yang digunakan untuk mengumpulkan data, training, evaluasi, hingga testing memiliki spesifikasi Intel(R) Core(TM) i7-8565U CPU @1.80GHz 1.99 GHz, 16 GB LPDDR3 RAM, 512GB M.2 NVMe PCIe SSD, NVIDIA GeForce MX150 with 2GB GDDR5 VRAM, Windows 10 Pro 64-bit.
2. GoogleColab merupakan produk Google research berbasis Cloud yang dapat digunakan secara gratis. Untuk menjalankan training model sistem klasifikasi menggunakan IndRNN dibutuhkan TPU serta *Runtime Shape High-RAM* karena jumlah data yang ditraining cukup besar.
3. PyCharm edisi 2021.3.3 digunakan untuk melakukan ekstraksi data secara lokal. PyCharm merupakan IDE (*Integrated Development Environment*) yang biasa digunakan un-

tuk membangun model *machine learning* yang menawarkan kemudahan untuk membangun *environment* berbeda sesuai kebutuhan.

## **BAB IV**

### **HASIL DAN PEMBAHASAN**

Pada penelitian ini akan dipaparkan hasil dan pembahasan dari proses-proses yang telah dilakukan pada bab metodologi. Hasil pembahasan yang ada pada bab ini diharapkan agar dapat ditarik suatu kesimpulan dari pelaksanaan tugas akhir yang telah dilakukan.

#### **4.1 Hasil Pengumpulan Data**

Video yang digunakan sebagai masukan dari model yang dibuat, diambil dari dataset UTA-RLDD. Banyak video yang perlu disaring pada dataset ini guna menghasilkan data yang cukup baik. Video dengan fps yang mendekati 30 fps saja yang dapat digunakan sebagai data masukan model. Didapatkan sebanyak 42 video pada setiap kelas seperti gambar 4.1 sehingga jumlah video yang dapat diekstraksi datanya berjumlah 126 video. Video yang ada pada dataset ini terbagi menjadi 3 kelas berbeda yaitu kelas 1 (Waspada), kelas 2 (Kewaspadaan Rendah), serta kelas 3 (Mengantuk). Subjek pada video yang diambil terdiri dari pria dan wanita yang berasal dari berbagai etnis serta sejumlah subjek yang ada pada video memakai kacamata serta memiliki bulu yang lebat di wajah. Berikut merupakan gambar sebaran jumlah video berdasarkan masing-masing kelas:

Tabel 4.1: Sebaran Video Tiap Kelas

Kelas	Jumlah Video
1	42
2	42
3	42

#### **4.2 Hasil Ekstraksi Fitur**

Ekstraksi fitur dilakukan secara lokal dengan bantuan IDE PyCharm edisi 2021.3.3 menghasilkan koordinat setiap titik pada mata maupun mulut berupa sumbu (x, y). Titik-titik koordinat tersebut nantinya akan digunakan untuk melakukan perhitungan sesuai dengan rumus 2.1 untuk menghasilkan nilai EAR serta rumus 2.2 untuk menghasilkan nilai MAR. Berikut merupakan gambaran data-data yang didapat setelah melakukan ekstraksi fitur:

Tabel 4.2: Deskripsi Data Hasil Ekstraksi Fitur

No	Nama Kolom	Tipe Data	Deskripsi
1	frame	<i>integer</i>	Jumlah Frame
2	timestamp_x	<i>float</i>	Waktu (detik)
3	xl0_x	<i>float</i>	Titik x pada koordinat 36 Landmark Wajah
4	yl0_x	<i>float</i>	Titik y pada koordinat 36 Landmark Wajah
5	xl1_x	<i>float</i>	Titik x pada koordinat 37 Landmark Wajah
6	yl1_x	<i>float</i>	Titik y pada koordinat 37 Landmark Wajah

No	Nama Kolom	Tipe Data	Deskripsi
7	xl2_x	float	Titik x pada koordinat 38 Landmark Wajah
8	yl2_x	float	Titik y pada koordinat 38 Landmark Wajah
9	xl3_x	float	Titik x pada koordinat 39 Landmark Wajah
10	yl3_x	float	Titik y pada koordinat 39 Landmark Wajah
11	xl4_x	float	Titik x pada koordinat 40 Landmark Wajah
12	yl4_x	float	Titik y pada koordinat 40 Landmark Wajah
13	xl5_x	float	Titik x pada koordinat 41 Landmark Wajah
14	yl5_x	float	Titik y pada koordinat 41 Landmark Wajah
15	xr0_x	float	Titik x pada koordinat 42 Landmark Wajah
16	yr0_x	float	Titik y pada koordinat 42 Landmark Wajah
17	xr1_x	float	Titik x pada koordinat 43 Landmark Wajah
18	yr1_x	float	Titik y pada koordinat 43 Landmark Wajah
19	xr2_x	float	Titik x pada koordinat 44 Landmark Wajah
20	yr2_x	float	Titik y pada koordinat 44 Landmark Wajah
21	xr3_x	float	Titik x pada koordinat 45 Landmark Wajah
22	yr3_x	float	Titik y pada koordinat 45 Landmark Wajah
23	xr4_x	float	Titik x pada koordinat 46 Landmark Wajah
24	yr4_x	float	Titik y pada koordinat 46 Landmark Wajah
25	xr5_x	float	Titik x pada koordinat 47 Landmark Wajah
26	yr5_x	float	Titik y pada koordinat 47 Landmark Wajah
27	ear_x	float	Nilai Kalkulasi EAR
28	x0_x	float	Titik x pada koordinat 60 Landmark Wajah
29	y0_x	float	Titik y pada koordinat 60 Landmark Wajah
30	x1_x	float	Titik x pada koordinat 61 Landmark Wajah
31	y1_x	float	Titik y pada koordinat 61 Landmark Wajah
32	x2_x	float	Titik x pada koordinat 62 Landmark Wajah
33	y2_x	float	Titik y pada koordinat 62 Landmark Wajah
34	x3_x	float	Titik x pada koordinat 63 Landmark Wajah
35	y3_x	float	Titik y pada koordinat 63 Landmark Wajah
36	x4_x	float	Titik x pada koordinat 64 Landmark Wajah
37	y4_x	float	Titik y pada koordinat 64 Landmark Wajah
38	x5_x	float	Titik x pada koordinat 65 Landmark Wajah
39	y5_x	float	Titik y pada koordinat 65 Landmark Wajah
40	x6_x	float	Titik x pada koordinat 66 Landmark Wajah
41	y6_x	float	Titik y pada koordinat 66 Landmark Wajah
42	x7_x	float	Titik x pada koordinat 67 Landmark Wajah
43	y7_x	float	Titik y pada koordinat 67 Landmark Wajah
44	mar_x	float	Nilai Kalkulasi MAR

### 4.3 Hasil Preprocessing

Preprocessing data dilakukan untuk menyiapkan data pelatihan model sehingga dapat menghasilkan model klasifikasi yang baik. Berikut hasil implementasikan praproses data pada pengembangan penelitian diantaranya melakukan ekstraksi fitur, mengatasi data/frame kosong, melakukan penyetaraan jumlah data dan *labeling*, melakukan normalisasi data, serta melakukan

penyesuaian bentuk data.

### 4.3.1 Hasil Interpolasi Data

Video yang telah terpilih tidak selalu memiliki data yang lengkap pada tiap frame. Terdapat data yang bernilai NaN dikarenakan koordinat landmark wajahnya tidak terdeteksi. Oleh karena itu digunakan interpolasi linear untuk mengisi data yang nilainya NaN sehingga titik-titik koordinat yang dibutuhkan untuk menghitung nilai EAR maupun MAR menjadi lengkap dan bisa mendapatkan nilai dari EAR maupun MAR itu sendiri. Berikut merupakan contoh tabel data yang memiliki nilai NaN di dalamnya sebelum maupun setelah dilakukannya interpolasi:

Tabel 4.3: Data Kosong Sebelum Interpolasi

timestamp_x	xl0_x	...	mar_x
10.329	78	...	0.1197736075
10.362	78	...	0.1427218439
10.395	NaN	...	NaN
10.428	NaN	...	NaN
10.461	77	...	0.1290500622
10.494	77	...	0.1433720878
10.527	78	...	0.1215658232

Tabel 4.4: Data Kosong Setelah Interpolasi

timestamp_x	xl0_x	...	mar_x
10.329	78	...	0.1197736075
10.362	78	...	0.1427218439
10.395	78	...	0.05930266631
10.428	77	...	0.05930266631
10.461	77	...	0.1290500622
10.494	77	...	0.1433720878
10.527	78	...	0.1215658232

### 4.3.2 Hasil Data Balancing dan Labeling

Panjang data yang didapatkan pada setiap video tidaklah sama seperti pada tabel 4.5, oleh karena itu perlu dilakukan penyetaraan panjang data pada tiap video sehingga dapat menjadi inputan untuk model klasifikasi. Pada tahap ini hanya kolom ear\_x dan mar\_x saja yang digunakan. Panjang data yang dibutuhkan adalah 16200 data tiap video. Setelah jumlah data diseragamkan, maka data-data tersebut akan diberi label tiap frame sesuai dengan label yang telah ditentukan oleh dataset UTA-RLDD. Berikut merupakan tabel rincian jumlah data yang didapat yang ada pada setiap video:

Tabel 4.5: Jumlah Data Pada Tiap Video

No	Video	Data	No	Video	Data	No	Video	Data
1	13-1.csv	18234	43	10-2.csv	18620	85	10-3.csv	18866
2	35-1.csv	18027	44	13-2.csv	20477	86	13-3.csv	18486

No	Video	Data	No	Video	Data	No	Video	Data
3	59-1.csv	18089	45	59-2.csv	18169	87	34-3.csv	18388
4	7-1.csv	18016	46	19-2.csv	18035	88	35-3.csv	18340
5	21-1.csv	18151	47	21-2.csv	18031	89	59-3.csv	18095
6	19-1.csv	18029	48	9-2.csv	18367	90	7-3.csv	17939
7	9-1.csv	18172	49	34-2.csv	20693	91	20-3.csv	21437
8	60-1.csv	18054	50	60-2.csv	17533	92	19-3.csv	18064
9	42-1.csv	18021	51	35-2.csv	18022	93	21-3.csv	18055
10	5-1.csv	18410	52	56-2.csv	18354	94	9-3.csv	19816
11	8-1.csv	18158	53	42-2.csv	18122	95	60-3.csv	18581
12	10-1.csv	16417	54	40-2.csv	18036	96	5-3.csv	18050
13	56-1.csv	18355	55	36-2.csv	18792	97	8-3.csv	17989
14	14-1.csv	18386	56	5-2.csv	18062	98	56-3.csv	19417
15	15-1.csv	19301	57	7-2.csv	17886	99	14-3.csv	18692
16	22-1.csv	18221	58	14-2.csv	19742	100	15-3.csv	18483
17	1-1.csv	18055	59	15-2.csv	18541	101	22-3.csv	18316
18	2-1.csv	18263	60	22-2.csv	18266	102	1-3.csv	18012
19	17-1.csv	18121	61	1-2.csv	18033	103	2-3.csv	18124
20	12-1.csv	16919	62	2-2.csv	18106	104	3-3.csv	18480
21	23-1.csv	17567	63	8-2.csv	18048	105	17-3.csv	18212
22	20-1.csv	19122	64	3-2.csv	18514	106	23-3.csv	18187
23	25-1.csv	17992	65	17-2.csv	18245	107	25-3.csv	17987
24	26-1.csv	18489	66	12-2.csv	18838	108	26-3.csv	20849
25	27-1.csv	19724	67	20-2.csv	17957	109	27-3.csv	18627
26	30-1.csv	20439	68	23-2.csv	17286	110	30-3.csv	18251
27	36-1.csv	18694	69	25-2.csv	18016	111	36-3.csv	18728
28	37-1.csv	20483	70	26-2.csv	26445	112	37-3.csv	20396
29	38-1.csv	18056	71	27-2.csv	18774	113	39-3.csv	18158
30	39-1.csv	19376	72	30-2.csv	20846	114	40-3.csv	18029
31	40-1.csv	18060	73	37-2.csv	18551	115	41-3.csv	25166
32	41-1.csv	18839	74	38-2.csv	18097	116	42-3.csv	18030
33	43-1.csv	35818	75	39-2.csv	18173	117	43-3.csv	34975
34	48-1.csv	18913	76	43-2.csv	35936	118	46-3.csv	17773
35	47-1.csv	18027	77	48-2.csv	18183	119	47-3.csv	18340
36	58-1.csv	18054	78	47-2.csv	18022	120	54-3.csv	18944
37	54-1.csv	19107	79	58-2.csv	18302	121	58-3.csv	18244
38	49-1.csv	18191	80	49-2.csv	17917	122	50-3.csv	20013
39	50-1.csv	18174	81	50-2.csv	21683	123	51-3.csv	18494
40	51-1.csv	18182	82	51-2.csv	18364	124	52-3.csv	18077
41	52-1.csv	18044	83	52-2.csv	18062	125	38-3.csv	18074
42	57-1.csv	18938	84	57-2.csv	18221	126	57-3.csv	18368

### 4.3.3 Hasil Normalisasi Data

Normalisasi data Z-score dilakukan dengan menggunakan fungsi Standard Scaler pada library sklearn. Rumus yang digunakan untuk mencari Z-score dari setiap data adalah rumus 2.4. Cara kerjanya sendiri yaitu data akan dikurangi dengan nilai rata-rata dari keseluruhan data

kemudian dibagi oleh standard deviasi dari keseluruhan data. Normalisasi ini menyebabkan semua fitur memiliki skala yang serupa, sehingga memudahkan perbandingan dan interpretasi data. Berikut merupakan data sebelum dan sesudah dinormalisasi:

Tabel 4.6: Data Sebelum Normalisasi

	<b>ear_x</b>	<b>mar_x</b>	<b>label</b>
0	0.296401	0.067902	1
1	0.309457	0.054692	1
2	0.289860	0.068918	1
3	0.289045	0.044270	1
4	0.299951	0.046012	1
...	...	...	...
2041199	0.259205	0.176349	3

Tabel 4.7: Data Setelah Normalisasi

	<b>ear_x</b>	<b>mar_x</b>	<b>label</b>
0	0.166568	0.544814	1
1	0.399185	0.270716	1
2	0.050017	0.565894	1
3	0.035485	0.054473	1
4	0.229822	0.090614	1
...	...	...	...
2041199	-0.496194	2.794994	3

Setelah data dinormlisasi, data kemudian dibagi menjadi data untuk train sebanyak 101 video dan data test atau val sebanyak 25 video. Data train kemudian dilakukan augmentasi hingga menjadi 353 video untuk menghindari overfitting pada model.

#### 4.3.4 Hasil Sliding Window

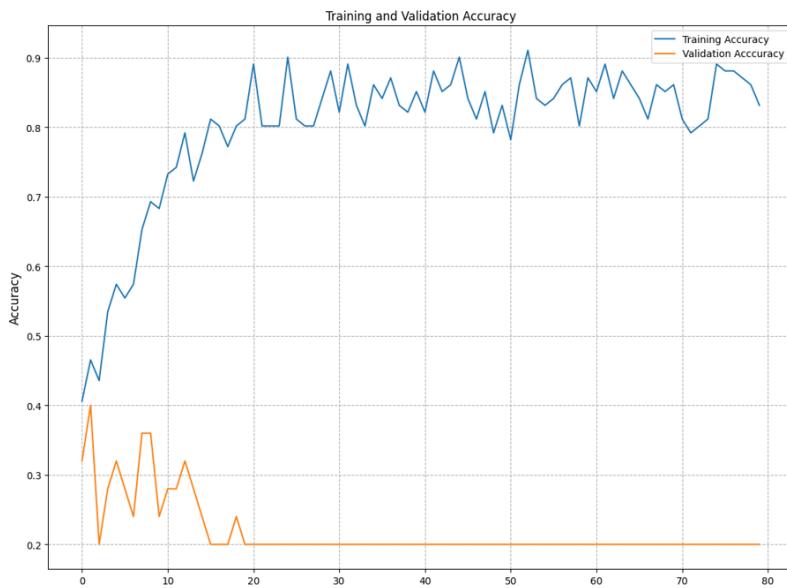
Bentuk data masukan yang dimiliki adalah 2 dimensi dimana bentuk data training yaitu (5718600, 2) dan bentuk data test/validasi yaitu (405000, 2) sedangkan yang dibutuhkan pada model IndRNN berjumlah 3 dimensi sehingga sliding window diperlukan agar bentuk data training menjadi (353, 16200, 2) dan bentuk data test/validasi (25, 16200, 2). Bentuk data tersebut dapat dijelaskan menjadi angka pertama merupakan jumlah video, 16200 merupakan jumlah langkah waktu / frame dalam satu video, serta 2 merupakan jumlah fitur yang digunakan yaitu EAR dan MAR.

### 4.4 Hasil *Data Training*

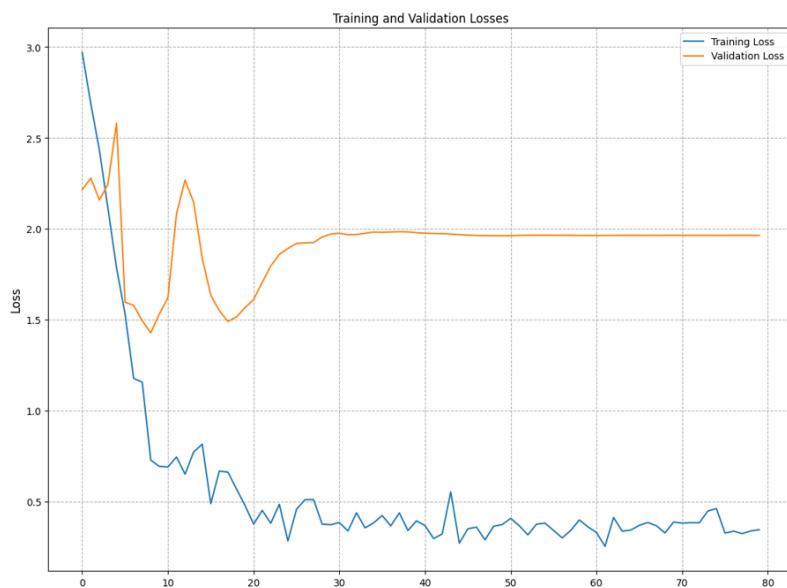
Hasil yang didapatkan setelah melakukan *Data Training* pada model yang dibuat dengan menggunakan arsitektur IndRNN, LSTM, dan SVM dapat dijabarkan sebagai berikut.

#### 4.4.1 Independent Recurrent Neural Network

Berikut merupakan gambar loss dan akurasi dari data training yang tidak teraugmentasi dan data validasi yang terbaik:

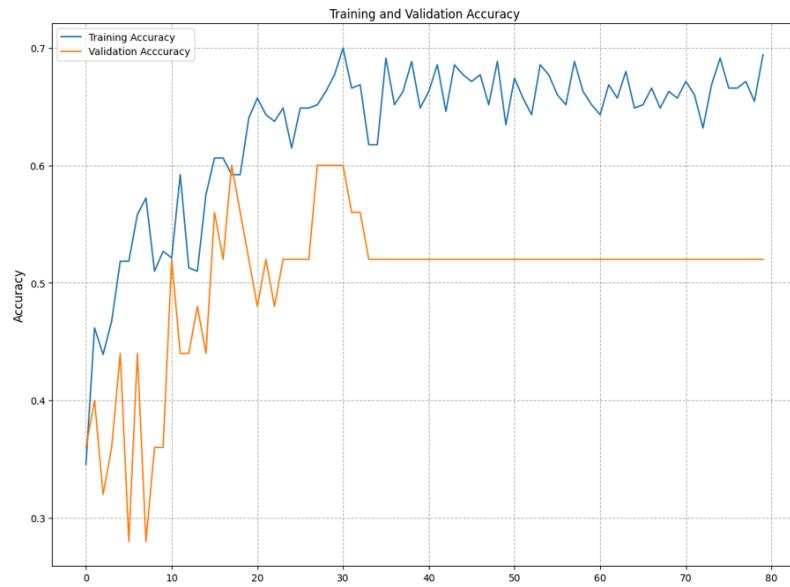


Gambar 4.1: Akurasi IndRNN Tanpa Data Augmentasi

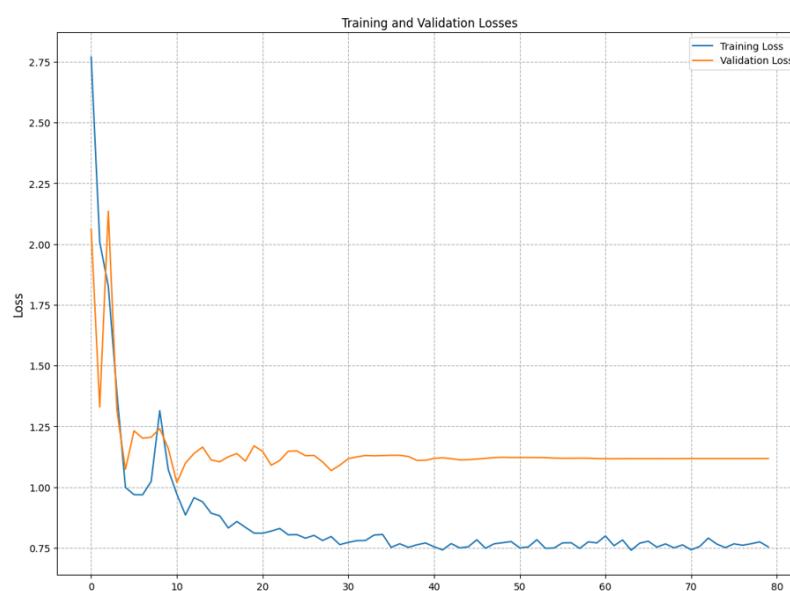


Gambar 4.2: Loss IndRNN Tanpa Data Augmentasi

Berikut merupakan gambar loss dan akurasi dari data training yang teraugmentasi dan data validasi yang terbaik:



Gambar 4.3: Akurasi dan Loss IndRNN Dengan Data Augmentasi

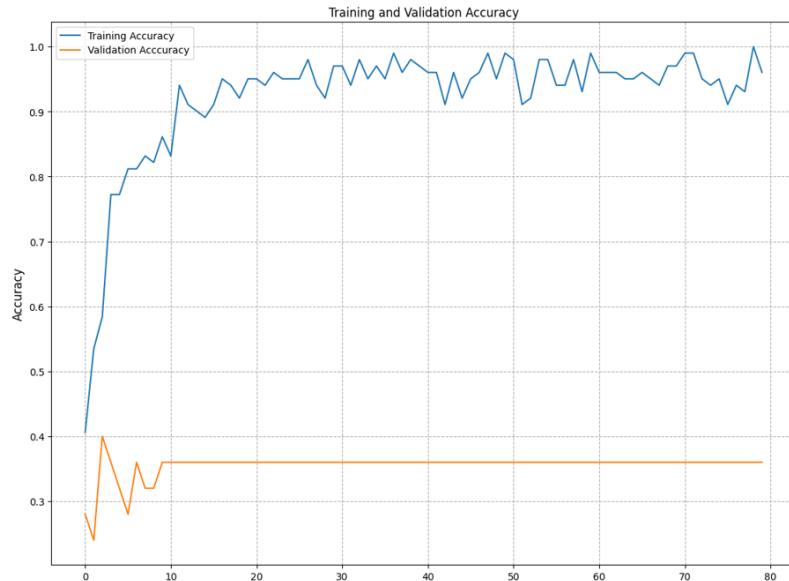


Gambar 4.4: Akurasi dan Loss IndRNN Dengan Data Augmentasi

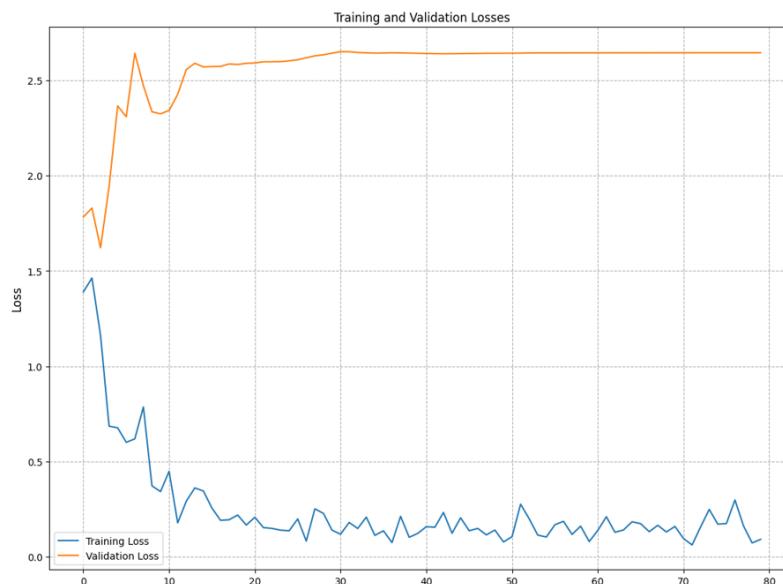
Data pada gambar 4.1 dan gambar 4.2 menunjukkan akurasi training sebesar 0.8378 dan akurasi validasi sebesar 0.2 serta loss training sebesar 0.3236 dan loss validasi sebesar 1.9798 . Sedangkan data pada gambar 4.3 dan gambar 4.4 menunjukkan akurasi training sebesar 0.6943 dan akurasi validasi sebesar 0.52 serta loss training sebesar 0.7529 dan loss validasi sebesar 1.1258 .

#### 4.4.2 Long short term memory network

Berikut merupakan gambar loss dan akurasi dari data training yang tidak teraugmentasi dan data validasi yang terbaik:

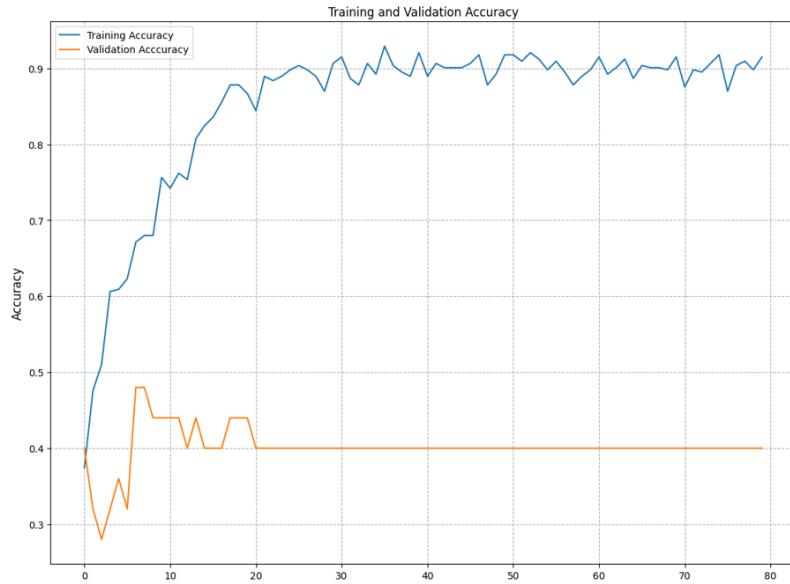


Gambar 4.5: Akurasi dan Loss LSTM Tanpa Data Augmentasi

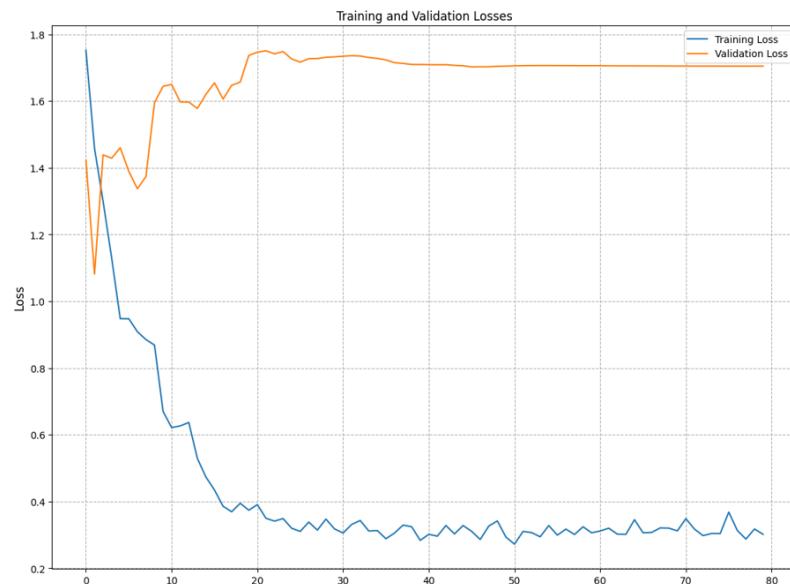


Gambar 4.6: Akurasi dan Loss LSTM Tanpa Data Augmentasi

Berikut merupakan gambar loss dan akurasi dari data training yang teraugmentasi dan data validasi yang terbaik:



Gambar 4.7: Akurasi dan Loss LSTM Dengan Data Augmentasi



Gambar 4.8: Akurasi dan Loss LSTM Dengan Data Augmentasi

Data pada gambar 4.5 dan gambar 4.6 menunjukkan akurasi training sebesar 0.9732 dan akurasi validasi sebesar 0.32 serta loss training sebesar 0.1342 dan loss validasi sebesar 2.4947 . Sedangkan data pada gambar 4.7 dan gambar 4.8 menunjukkan akurasi training sebesar 0.9354 dan akurasi validasi sebesar 0.4 serta loss training sebesar 0.3541 dan loss validasi sebesar 1.7487 .

## 4.5 Hasil Evaluasi Model

Hasil evaluasi yang dihasilkan pada penelitian ini berbentuk:

### 1. Confusion Matrix

Confusion matrix adalah tabel yang menunjukkan jumlah prediksi yang benar dan salah yang dilakukan oleh model. Confusion matrix terdiri dari empat komponen: *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Dengan menggunakan confusion matrix, kita dapat menghitung metrik evaluasi lainnya.

### 2. Akurasi

Akurasi mengukur persentase prediksi yang benar secara keseluruhan. Ini dihitung dengan membagi jumlah prediksi yang benar (TP dan TN) dengan total sampel dalam dataset. Akurasi dapat memberikan gambaran umum tentang sejauh mana model dapat memprediksi dengan benar, tetapi dapat menghasilkan hasil yang bias jika terdapat ketimpangan jumlah sampel pada setiap kelas

### 3. Presisi

Presisi mengukur persentase prediksi positif yang benar. Ini mengukur sejauh mana prediksi positif model adalah akurat. Presisi dihitung dengan membagi TP dengan jumlah prediksi positif (TP dan FP).

### 4. Recall

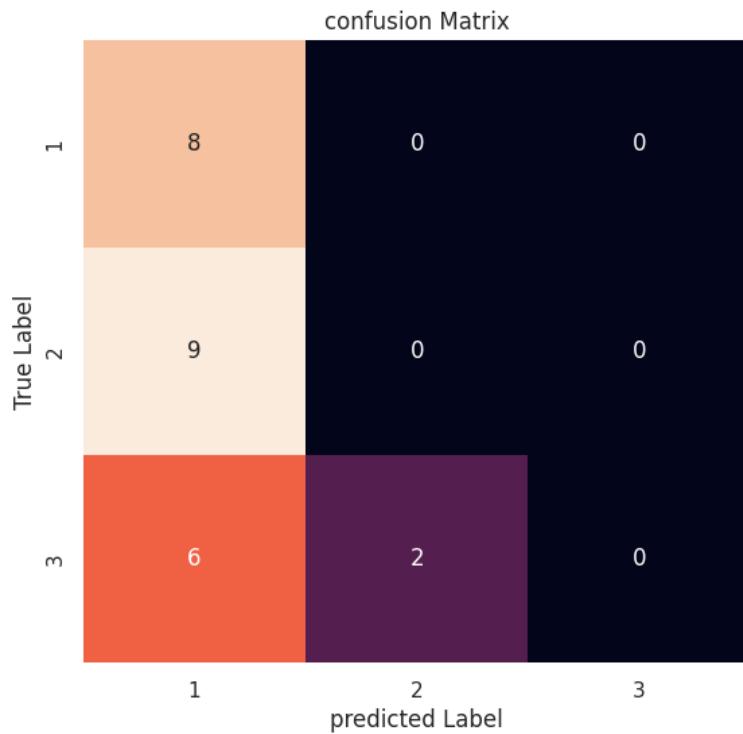
Recall, juga dikenal sebagai *Sensitivity* atau *True Positive Rate* (TPR), mengukur persentase sampel positif yang diprediksi dengan benar. Ini mengukur sejauh mana model dapat menemukan semua sampel positif yang sebenarnya. Recall dihitung dengan membagi TP dengan jumlah sampel positif (TP dan FN).

### 5. F1-Score

F1-Score adalah penggabungan presisi dan recall menjadi satu metrik yang mencerminkan keseimbangan antara keduanya. F1-Score dihitung sebagai rata-rata harmonis dari presisi dan recall, dan memberikan gambaran keseluruhan tentang performa model. F1-Score berguna ketika kelas target memiliki ketidakseimbangan jumlah sampel.

### 4.5.1 Independent Recurrent Neural Network

Berikut merupakan gambar hasil evaluasi model IndRNN tanpa data augmentasi:



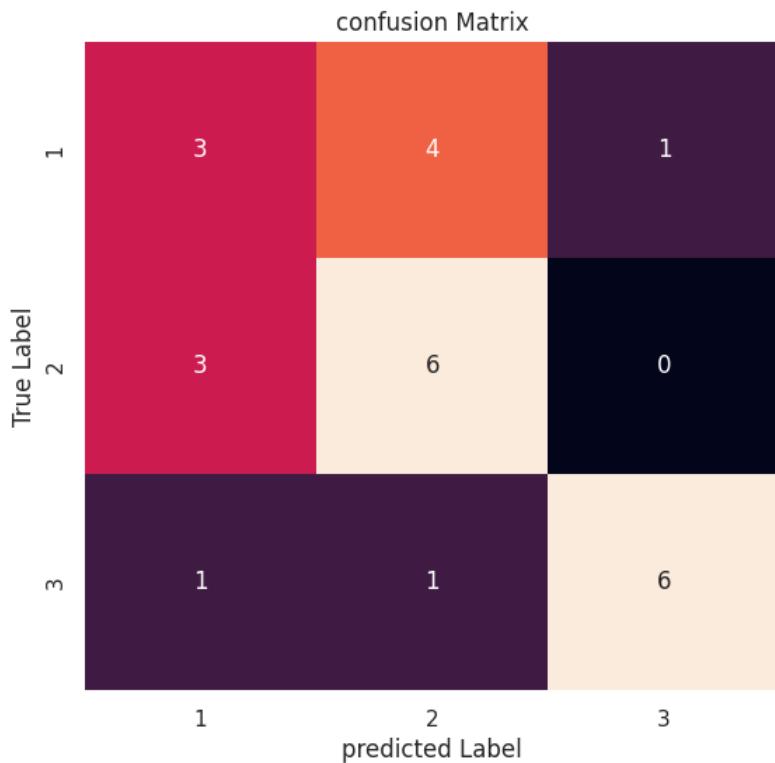
Gambar 4.9: Confusion Matrix IndRNN Tanpa Data Augmentasi

	precision	recall	f1-score	support
0	0.35	1.00	0.52	8
1	0.00	0.00	0.00	9
2	0.00	0.00	0.00	8
accuracy			0.32	25
macro avg	0.12	0.33	0.17	25
weighted avg	0.11	0.32	0.17	25

Gambar 4.10: Evaluasi IndRNN Tanpa Data Augmentasi

Data yang didapatkan dari confusion matrix 4.9 didapatkan 8 video yang bernilai benar dengan hasil akurasi sebesar 0.32, hasil nilai presisi model pada kelas 1 sebesar 0.35, hasil nilai presisi model pada kelas 2 sebesar 0.0, hasil nilai presisi model pada kelas 3 sebesar 0.0. Model menghasilkan nilai recall sebesar 1.0 pada kelas 1, 0.0 pada kelas 2, dan 0.0 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.52 pada kelas 1, 0.0 pada kelas 2, 0.0 pada kelas 3.

Berikut merupakan gambar hasil evaluasi model IndRNN dengan data augmentasi:



Gambar 4.11: Confusion Matrix IndRNN Dengan Data Augmentasi

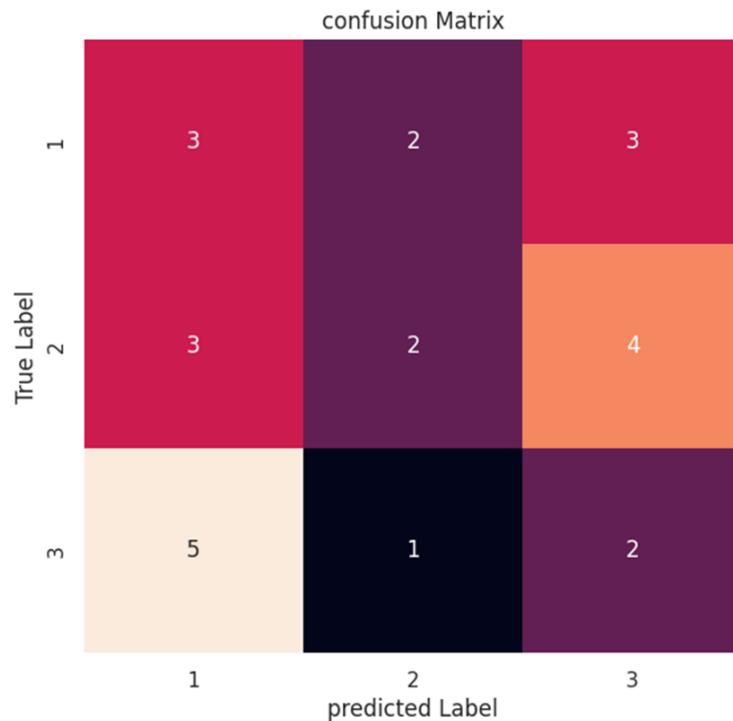
	precision	recall	f1-score	support
0	0.43	0.38	0.40	8
1	0.55	0.67	0.60	9
2	0.86	0.75	0.80	8
accuracy			0.60	25
macro avg	0.61	0.60	0.60	25
weighted avg	0.61	0.60	0.60	25

Gambar 4.12: Evaluasi IndRNN Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.11 didapatkan 15 video yang bernilai benar dengan hasil akurasi sebesar 0.6, hasil nilai presisi model pada kelas 1 sebesar 0.43, hasil nilai presisi model pada kelas 2 sebesar 0.55, hasil nilai presisi model pada kelas 3 sebesar 0.86. Model menghasilkan nilai recall sebesar 0.38 pada kelas 1, 0.67 pada kelas 2, dan 0.75 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.40 pada kelas 1, 0.60 pada kelas 2, 0.80 pada kelas 3.

#### 4.5.2 Long short term memory network

Berikut merupakan gambar hasil evaluasi model LSTM tanpa data augmentasi:



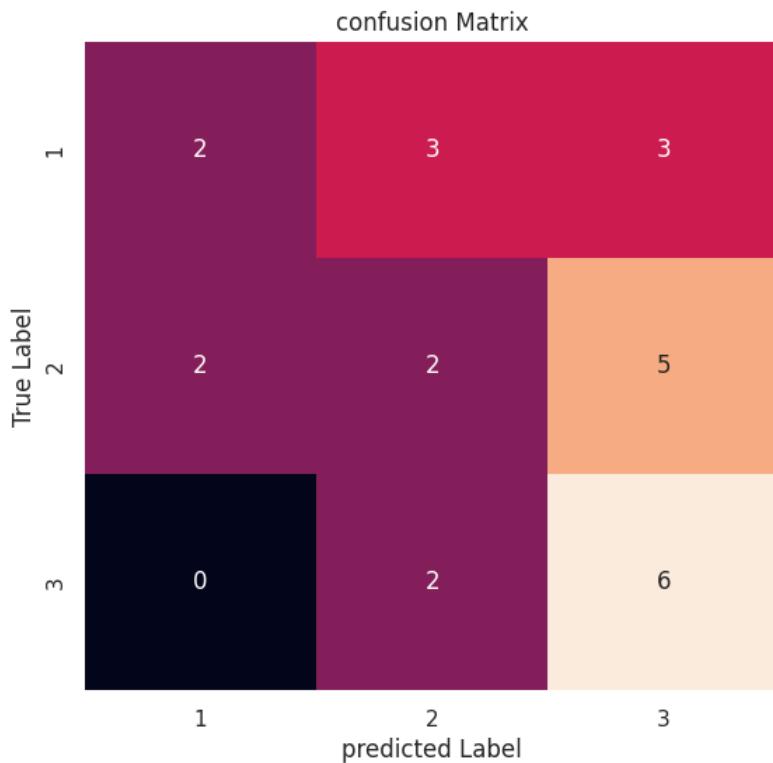
Gambar 4.13: Akurasi dan Loss LSTM Tanpa Data Augmentasi

	<b>precision</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
0	<b>0.27</b>	<b>0.38</b>	<b>0.32</b>	8
1	<b>0.40</b>	<b>0.22</b>	<b>0.29</b>	9
2	<b>0.22</b>	<b>0.25</b>	<b>0.24</b>	8
<b>accuracy</b>			<b>0.28</b>	25
<b>macro avg</b>	<b>0.30</b>	<b>0.28</b>	<b>0.28</b>	25
<b>weighted avg</b>	<b>0.30</b>	<b>0.28</b>	<b>0.28</b>	25

Gambar 4.14: Evaluasi LSTM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.13 didapatkan 7 video yang bernilai benar dengan hasil akurasi sebesar 0.28, hasil nilai presisi model pada kelas 1 sebesar 0.27, hasil nilai presisi model pada kelas 2 sebesar 0.4, hasil nilai presisi model pada kelas 3 sebesar 0.22. Model menghasilkan nilai recall sebesar 0.38 pada kelas 1, 0.22 pada kelas 2, dan 0.25 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.32 pada kelas 1, 0.29 pada kelas 2, 0.24 pada kelas 3.

Berikut merupakan gambar hasil evaluasi model LSTM dengan data augmentasi:



Gambar 4.15: Confusion Matrix LSTM Dengan Data Augmentasi

↳	precision	recall	f1-score	support
0	0.50	0.25	0.33	8
1	0.29	0.22	0.25	9
2	0.43	0.75	0.55	8
accuracy			0.40	25
macro avg	0.40	0.41	0.38	25
weighted avg	0.40	0.40	0.37	25

Gambar 4.16: Evaluasi LSTM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.15 didapatkan 10 video yang bernilai benar dengan hasil akurasi sebesar 0.40, hasil nilai presisi model pada kelas 1 sebesar 0.5, hasil nilai presisi model pada kelas 2 sebesar 0.29, hasil nilai presisi model pada kelas 3 sebesar 0.43. Model menghasilkan nilai recall sebesar 0.25 pada kelas 1, 0.22 pada kelas 2, dan 0.275 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.33 pada kelas 1, 0.25 pada kelas 2, 0.55 pada kelas 3.

### 4.5.3 Support Vector Machine

Berikut merupakan gambar hasil evaluasi model SVM tanpa data augmentasi:

$$\begin{bmatrix} \begin{bmatrix} 3 & 1 & 4 \end{bmatrix} \\ \begin{bmatrix} 4 & 3 & 2 \end{bmatrix} \\ \begin{bmatrix} 2 & 3 & 3 \end{bmatrix} \end{bmatrix}$$

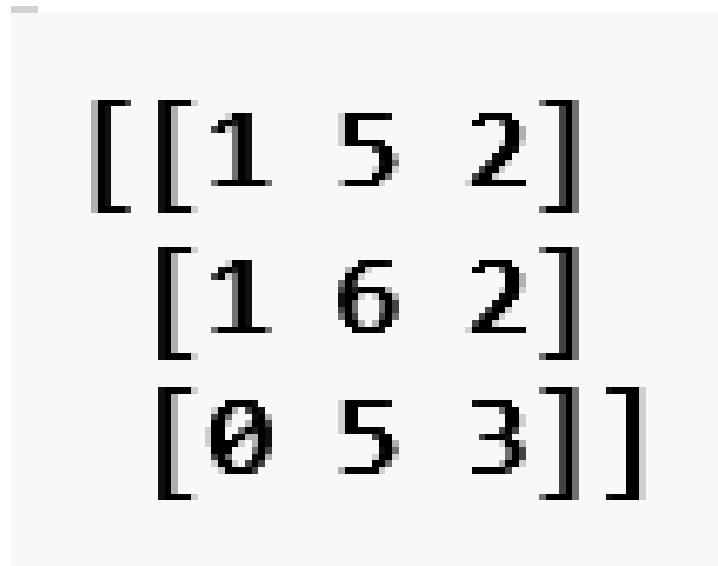
Gambar 4.17: Confusion Matrix SVM Tanpa Data Augmentasi

	<b>precision</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
1	0.33	0.38	0.35	8
2	0.43	0.33	0.38	9
3	0.33	0.38	0.35	8
<b>accuracy</b>			0.36	25
<b>macro avg</b>	0.37	0.36	0.36	25
<b>weighted avg</b>	0.37	0.36	0.36	25

Gambar 4.18: Evaluasi SVM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.17 didapatkan 9 video yang bernilai benar dengan hasil akurasi sebesar 0.36, hasil nilai presisi model pada kelas 1 sebesar 0.33, hasil nilai presisi model pada kelas 2 sebesar 0.43, hasil nilai presisi model pada kelas 3 sebesar 0.33. Model menghasilkan nilai recall sebesar 0.38 pada kelas 1, 0.33 pada kelas 2, dan 0.38 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.35 pada kelas 1, 0.38 pada kelas 2, 0.35 pada kelas 3.

Berikut merupakan gambar hasil evaluasi model SVM dengan data augmentasi:



Gambar 4.19: Confusion Matrix SVM Dengan Data Augmentasi

	precision	recall	f1-score	support
1	0.50	0.12	0.20	8
2	0.38	0.67	0.48	9
3	0.43	0.38	0.40	8
accuracy			0.40	25
macro avg	0.43	0.39	0.36	25
weighted avg	0.43	0.40	0.36	25

Gambar 4.20: Evaluasi SVM Dengan Data Augmentasi

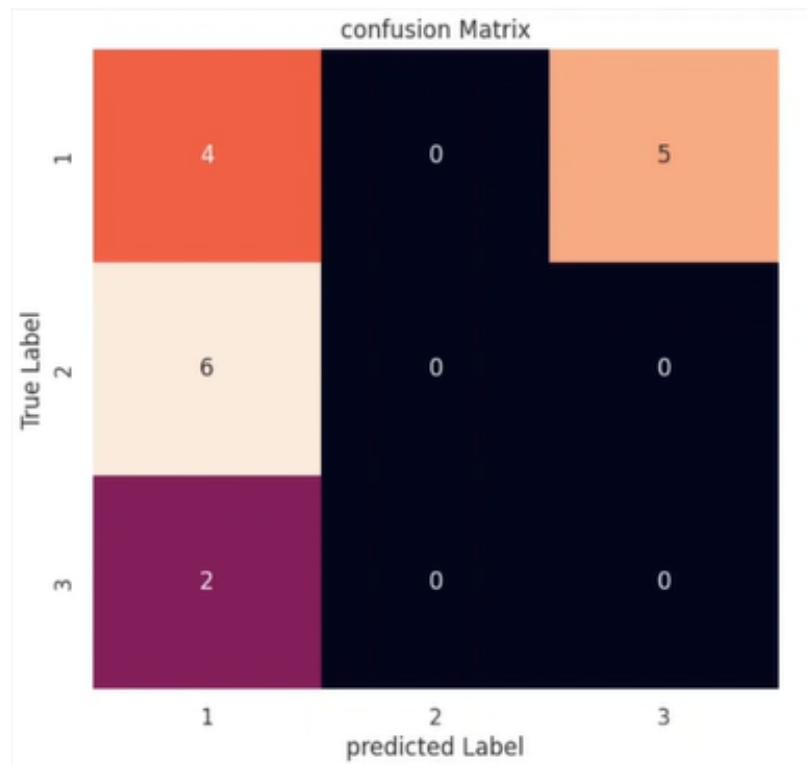
Data yang didapatkan dari confusion matrix 4.19 didapatkan 10 video yang bernilai benar dengan hasil akurasi sebesar 0.40, hasil nilai presisi model pada kelas 1 sebesar 0.50, hasil nilai presisi model pada kelas 2 sebesar 0.38, hasil nilai presisi model pada kelas 3 sebesar 0.43. Model menghasilkan nilai recall sebesar 0.12 pada kelas 1, 0.67 pada kelas 2, dan 0.38 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.20 pada kelas 1, 0.48 pada kelas 2, 0.40 pada kelas 3.

## 4.6 Hasil Pengujian Model

Hasil yang didapatkan setelah melakukan pengujian model pada model yang dibuat dengan menggunakan arsitektur IndRNN, LSTM, dan SVM dapat dijabarkan sebagai berikut.

### 4.6.1 Independent Recurrent Neural Network

Berikut merupakan gambar hasil evaluasi model IndRNN tanpa data augmentasi:



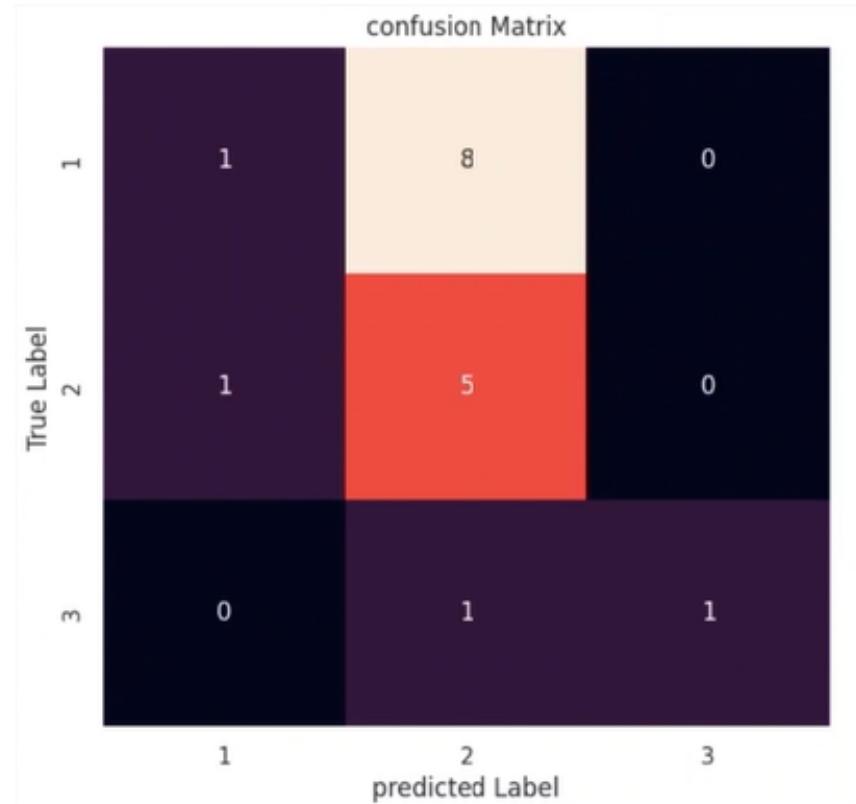
Gambar 4.21: Confusion Matrix IndRNN Tanpa Data Augmentasi

	<b>precision</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
0	<b>0.33</b>	<b>0.44</b>	<b>0.38</b>	9
1	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	6
2	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	2
<b>accuracy</b>			<b>0.24</b>	17
<b>macro avg</b>	<b>0.11</b>	<b>0.15</b>	<b>0.13</b>	17
<b>weighted avg</b>	<b>0.18</b>	<b>0.24</b>	<b>0.20</b>	17

Gambar 4.22: Testing IndRNN Tanpa Data Augmentasi

Data yang didapatkan dari confusion matrix 4.21 didapatkan 4 video yang bernilai benar dengan hasil akurasi sebesar 0.24, hasil nilai presisi model pada kelas 1 sebesar 0.33, hasil nilai presisi model pada kelas 2 sebesar 0.0, hasil nilai presisi model pada kelas 3 sebesar 0.0. Model menghasilkan nilai recall sebesar 0.44 pada kelas 1, 0.0 pada kelas 2, dan 0.0 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.38 pada kelas 1, 0.0 pada kelas 2, 0.0 pada kelas 3.

Berikut merupakan gambar hasil evaluasi model IndRNN dengan data augmentasi:



Gambar 4.23: Confusion Matrix IndRNN Dengan Data Augmentasi

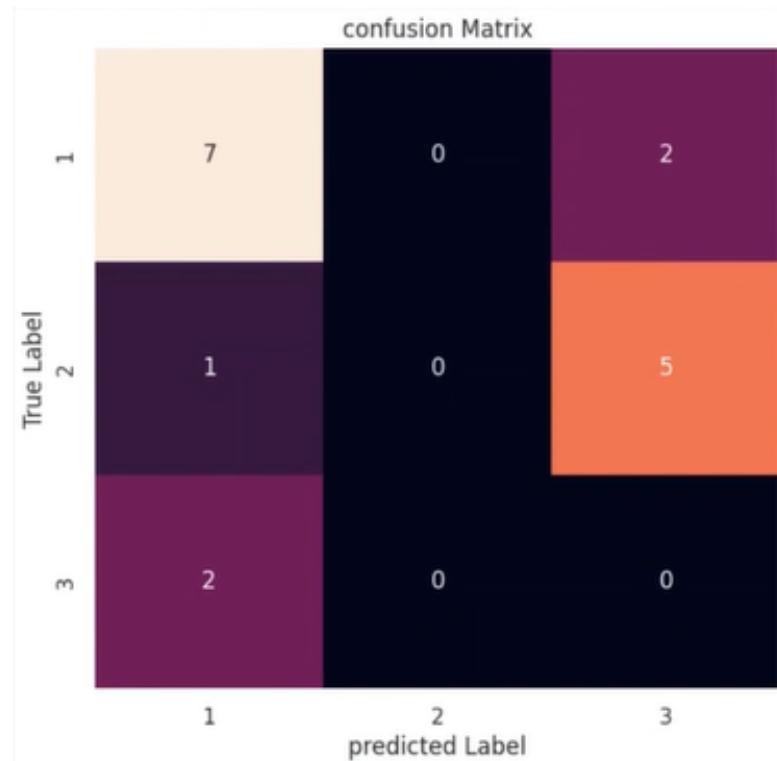
	<b>precision</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
<b>0</b>	<b>0.50</b>	<b>0.11</b>	<b>0.18</b>	<b>9</b>
<b>1</b>	<b>0.36</b>	<b>0.83</b>	<b>0.50</b>	<b>6</b>
<b>2</b>	<b>1.00</b>	<b>0.50</b>	<b>0.67</b>	<b>2</b>
<b>accuracy</b>			<b>0.41</b>	<b>17</b>
<b>macro avg</b>	<b>0.62</b>	<b>0.48</b>	<b>0.45</b>	<b>17</b>
<b>weighted avg</b>	<b>0.51</b>	<b>0.41</b>	<b>0.35</b>	<b>17</b>

Gambar 4.24: Testing IndRNN Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.23 didapatkan 7 video yang bernilai benar dengan hasil akurasi sebesar 0.41, hasil nilai presisi model pada kelas 1 sebesar 0.50, hasil nilai presisi model pada kelas 2 sebesar 0.36, hasil nilai presisi model pada kelas 3 sebesar 1.0. Model menghasilkan nilai recall sebesar 0.11 pada kelas 1, 0.83 pada kelas 2, dan 0.50 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.18 pada kelas 1, 0.50 pada kelas 2, 0.67 pada kelas 3.

#### 4.6.2 Long short term memory network

Berikut merupakan gambar hasil evaluasi model LSTM tanpa data augmentasi:



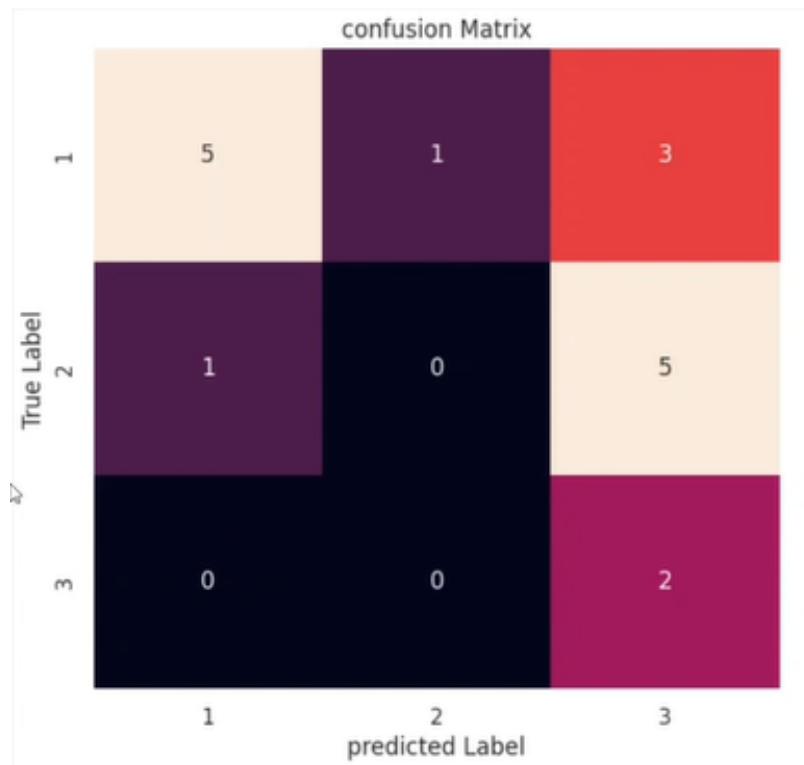
Gambar 4.25: Confusion Matrix LSTM Tanpa Data Augmentasi

	precision	recall	f1-score	support
0	0.70	0.78	0.74	9
1	0.00	0.00	0.00	6
2	0.00	0.00	0.00	2
accuracy			0.41	17
macro avg	0.23	0.26	0.25	17
weighted avg	0.37	0.41	0.39	17

Gambar 4.26: Testing LSTM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.25 didapatkan 7 video yang bernilai benar dengan hasil akurasi sebesar 0.41, hasil nilai presisi model pada kelas 1 sebesar 0.70, hasil nilai presisi model pada kelas 2 sebesar 0.0, hasil nilai presisi model pada kelas 3 sebesar 0.0. Model menghasilkan nilai recall sebesar 0.78 pada kelas 1, 0.0 pada kelas 2, dan 0.0 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.74 pada kelas 1, 0.0 pada kelas 2, 0.0 pada kelas 3.

Berikut merupakan gambar hasil evaluasi model LSTM dengan data augmentasi:



Gambar 4.27: Confusion Matrix LSTM Dengan Data Augmentasi

	<b>precision</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
0	<b>0.83</b>	<b>0.56</b>	<b>0.67</b>	9
1	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	6
2	<b>0.20</b>	<b>1.00</b>	<b>0.33</b>	2
<b>accuracy</b>			<b>0.41</b>	17
<b>macro avg</b>	<b>0.34</b>	<b>0.52</b>	<b>0.33</b>	17
<b>weighted avg</b>	<b>0.46</b>	<b>0.41</b>	<b>0.39</b>	17

Gambar 4.28: Testing LSTM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.27 didapatkan 7 video yang bernilai benar dengan hasil akurasi sebesar 0.41, hasil nilai presisi model pada kelas 1 sebesar 0.83, hasil nilai presisi model pada kelas 2 sebesar 0.0, hasil nilai presisi model pada kelas 3 sebesar 0.20. Model menghasilkan nilai recall sebesar 0.56 pada kelas 1, 0.0 pada kelas 2, dan 1.0 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.67 pada kelas 1, 0.0 pada kelas 2, 0.33 pada kelas 3.

#### 4.6.3 Support Vector Machine

Berikut merupakan gambar hasil evaluasi model SVM tanpa data augmentasi:

$$\begin{bmatrix} [5 & 1 & 3] \\ [1 & 1 & 4] \\ [0 & 1 & 1] \end{bmatrix}$$

Gambar 4.29: Confusion Matrix SVM Tanpa Data Augmentasi

C	precision	recall	f1-score	support
1	0.83	0.56	0.67	9
2	0.33	0.17	0.22	6
3	0.12	0.50	0.20	2
accuracy			0.41	17
macro avg	0.43	0.41	0.36	17
weighted avg	0.57	0.41	0.45	17

Gambar 4.30: Testing SVM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.29 didapatkan 7 video yang bernilai benar dengan hasil akurasi sebesar 0.41, hasil nilai presisi model pada kelas 1 sebesar 0.83, hasil nilai presisi model pada kelas 2 sebesar 0.33, hasil nilai presisi model pada kelas 3 sebesar 0.12. Model menghasilkan nilai recall sebesar 0.56 pada kelas 1, 0.17 pada kelas 2, dan 0.50 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.67 pada kelas 1, 0.22 pada kelas 2, 0.20 pada kelas 3.

Berikut merupakan gambar hasil evaluasi model SVM dengan data augmentasi:

<b>[</b>	<b>[</b>	<b>5</b>	<b>1</b>	<b>3</b>	<b>]</b>
<b>[</b>	<b>0</b>	<b>2</b>	<b>4</b>	<b>]</b>	<b>]</b>
<b>[</b>	<b>0</b>	<b>2</b>	<b>0</b>	<b>]</b>	<b>]</b>

Gambar 4.31: Confusion Matrix SVM Dengan Data Augmentasi

	<b>precision</b>	<b>recall</b>	<b>f1-score</b>	<b>support</b>
<b>1</b>	<b>1.00</b>	<b>0.56</b>	<b>0.71</b>	<b>9</b>
<b>2</b>	<b>0.40</b>	<b>0.33</b>	<b>0.36</b>	<b>6</b>
<b>3</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>2</b>
<b>accuracy</b>			<b>0.41</b>	<b>17</b>
<b>macro avg</b>	<b>0.47</b>	<b>0.30</b>	<b>0.36</b>	<b>17</b>
<b>weighted avg</b>	<b>0.67</b>	<b>0.41</b>	<b>0.51</b>	<b>17</b>

Gambar 4.32: Testing SVM Dengan Data Augmentasi

Data yang didapatkan dari confusion matrix 4.31 didapatkan 7 video yang bernilai benar dengan hasil akurasi sebesar 0.41, hasil nilai presisi model pada kelas 1 sebesar 1.0, hasil nilai presisi model pada kelas 2 sebesar 0.40, hasil nilai presisi model pada kelas 3 sebesar 0.0. Model menghasilkan nilai recall sebesar 0.56 pada kelas 1, 0.33 pada kelas 2, dan 0.0 pada kelas 3. Nilai f1-score yang dihasilkan oleh model sebesar 0.71 pada kelas 1, 0.36 pada kelas 2, 0.0 pada kelas 3.

## 4.7 Pembahasan

Dari pengujian yang telah dilakukan, berdasarkan hasil training kedua model yaitu dengan IndRNN dan LSTM didapatkan tabel sebagai berikut:

Tabel 4.8: Data Hasil Training Model

<b>Model</b>	<b>Akurasi</b>	<b>Validasi</b>	<b>Loss Akurasi</b>	<b>Loss Validasi</b>
IndRNN tanpa data augmentasi	0.8378	0.2	0.3236	1.9798
IndRNN dengan data augmentasi	0.6943	0.52	0.7529	1.1258
LSTM tanpa data augmentasi	0.9732	0.32	0.1342	2.4947
LSTM dengan data augmentasi	0.9354	0.4	0.3541	1.7487

Tabel 4.9: Data Hasil Evaluasi Model

Model	Confusion Matrix
IndRNN tanpa data augmentasi	8 data benar
IndRNN dengan data augmentasi	15 data benar
LSTM tanpa data augmentasi	7 data benar
LSTM dengan data augmentasi	10 data benar
SVM tanpa data augmentasi	9 data benar
SVM dengan data augmentasi	10 data benar

Berikut merupakan analisa pengujian yang didapatkan dari tabel 4.8 dan tabel 4.8.

Model yang pertama yaitu IndRNN dengan input (x) berupa nilai EAR serta MAR dan kelas sebagai target (y) mendapatkan hasil training yang kurang baik dengan akurasi sebesar 0.8378 dan validasi akurasi sebesar 0.2, pada grafik terlihat model mengalami overfitting.

Model yang kedua yaitu IndRNN dengan input (x) berupa nilai EAR serta MAR yang teraugmentasi dan kelas sebagai target (y) mendapatkan hasil training yang paling baik diantara model lainnya karena tidak mengalami overfitting dengan akurasi 0.6943 dan validasi akurasi 0.52.

Model yang ketiga yaitu LSTM dengan input (x) berupa nilai EAR serta MAR dan kelas sebagai target (y) mendapatkan hasil training yang kurang baik dengan akurasi sebesar 0.9732 dan validasi akurasi sebesar 0.32 , pada grafik terlihat model mengalami overfitting.

Model yang keempat yaitu LSTM dengan input (x) berupa nilai EAR serta MAR yang teraugmentasi dan kelas sebagai target (y) mendapatkan hasil training yang lebih baik daripada LSTM dengan data yang tidak teraugmentasi, selain itu pada grafik juga terlihat model mengalami overfitting dengan akurasi 0.9354 dan validasi akurasi 0.4.

Model yang kelima yaitu SVM dengan input (x) berupa nilai EAR serta MAR dan kelas sebagai target (y) mendapatkan hasil evaluasi yang lebih baik jika dibandingkan dengan IndRNN dan LSTM tanpa data augmentasi yang menghasilkan 9 nilai benar.

Model yang keenam yaitu SVM dengan input (x) berupa nilai EAR serta MAR yang teraugmentasi dan kelas sebagai target (y) mendapatkan hasil evaluasi yang lebih baik dari SVM dengan data yang tidak teraugmentasi dengan jumlah 10 nilai benar.

Berdasarkan keseluruhan hasil training dan evaluasi, didapatkan bahwa model klasifikasi terbaik yaitu model IndRNN yang ditraining menggunakan data yang telah diaugmentasi. Hal ini memvalidasi bahwa model IndRNN pada klasifikasi kantuk lebih baik daripada menggunakan SVM dan LSTM.

*[Halaman ini sengaja dikosongkan]*

## **BAB V**

### **KESIMPULAN DAN SARAN**

Bab ini menjelaskan mengenai kesimpulan yang telah didapat dari proses penelitian tugas akhir.

#### **5.1 Kesimpulan**

Pada Tugas Akhir ini telah dijelaskan pengembangan Sistem Klasifikasi Kantuk Berdasarkan Pola Kedipan Mata dan Pola Bukaan Mulut dengan IndRNN. Dari hasil pengujian yang telah dilakukan pada bab sebelumnya model klasifikasi kantuk IndRNN tanpa data training yang diaugmentasi menghasilkan akurasi sebesar 0.8378 dan akurasi validasi sebesar 0.2. Sedangkan model klasifikasi kantuk IndRNN yang menggunakan data training teraugmentasi menghasilkan akurasi sebesar 0.6943 dan akurasi validasi sebesar 0.52, lebih baik jika dibandingkan dengan kedua model yang lainnya karena cenderung tidak overfitting. Kedua model lain yang dibuat adalah model LSTM tanpa data training yang diaugmentasi menghasilkan akurasi sebesar 0.9732 dan akurasi validasi sebesar 0.32 dan model klasifikasi kantuk LSTM yang menggunakan data training teraugmentasi yang menghasilkan akurasi sebesar 0.9354 dan akurasi validasi sebesar 0.4 serta model SVM tanpa data training yang diaugmentasi menghasilkan akurasi sebesar 0.36 pada evaluasi model dan model SVM yang menggunakan data training teraugmentasi menghasilkan akurasi sebesar 0.40 pada evaluasi model.

Berdasarkan keseluruhan hasil evaluasi, didapatkan bahwa model klasifikasi terbaik yaitu model IndRNN yang ditraining menggunakan data yang telah diaugmentasi dengan menghasilkan 15 data benar jika dibandingkan dengan model LSTM dan SVM yang menghasilkan 10 data benar.

#### **5.2 Saran**

Untuk pengembangan lebih lanjut mengenai Tugas Akhir ini terdapat beberapa kemungkinan perbaikan yang dapat dilakukan untuk penyempurnaan implementasi model antara lain:

1. Mencoba model deteksi wajah yang berbeda karena saat menggunakan deteksi wajah dari library dlib didapati beberapa titik pada wajah tidak sesuai.
2. Melakukan metode preprocessing yang berbeda untuk menghasilkan data yang lebih baik.
3. Mencoba menggunakan hyperparameter yang berbeda, melakukan pengaturan training yang berbeda seperti jumlah epoch, learning rate, serta menggunakan fungsi callback yang berbeda.

*[Halaman ini sengaja dikosongkan]*

## DAFTAR PUSTAKA

- Ardinendradewi, Q., Setyaningsih, Y., & Kurniawan, B. (2022). Pengaruh karakteristik individu terhadap kelelahan pekerja pengolahan gudeg cv. x yogyakarta. *Jurnal Riset Kesehatan Masyarakat*, 2(2).
- Chung, H., & Shin, K.-s. (2018). Genetic algorithm-optimized long short-term memory network for stock market prediction. *Sustainability*, 10(10), 3765.
- Dedy Irawan, J., Handoko, F., & Adriantatri, E. (2019). Ruang kuliah pintar pemantau tingkat efektivitas pembelajaran yang dapat mendeteksi mahasiswa bosan dan mengantuk. *Seminar Nasional Inovasi dan Aplikasi Teknologi di Industri*.
- Dinges, D. F. (1995). An overview of sleepiness and accidents. *Journal of sleep research*, 4, 4–14.
- Farodisa, A. M. (2022). Klasifikasi skala kantuk karolinska berdasarkan nilai eye aspect ratio menggunakan deep learning.
- Ghoddoosian, R., Galib, M., & Athitsos, V. (2019). A realistic dataset and baseline temporal model for early drowsiness detection. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 0–0.
- Ibrahim, B. R., Khalifa, F. M., Zeebaree, S. R., Othman, N. A., Alkhayyat, A., Zebari, R. R., & Sadeq, M. A. (2021). Embedded system for eye blink detection using machine learning technique. *2021 1st Babylon International Conference on Information Technology and Science (BICITS)*, 58–62.
- Kazemi, V., & Sullivan, J. (2014). One millisecond face alignment with an ensemble of regression trees. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1867–1874.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436–444.
- Lepot, M., Aubin, J.-B., & Clemens, F. H. (2017). Interpolation in time series: An introductory overview of existing methods, their performance criteria and uncertainty assessment. *Water*, 9(10), 796.
- Li, S., Li, W., Cook, C., Zhu, C., & Gao, Y. (2018). Independently recurrent neural network (indrnn): Building a longer and deeper rnn. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5457–5466.
- Manaswi, N. K., & Manaswi, N. K. (2018). Rnn and lstm. *Deep Learning with Applications Using Python: Chatbots and Face, Object, and Speech Recognition With TensorFlow and Keras*, 115–126.
- Massoz, Q., Langohr, T., François, C., & Verly, J. G. (2016). The ulg multimodality drowsiness database (called drozy) and examples of use. *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1–7.
- Maulana, M. A. (2022). Deteksi kantuk pada pengendara roda empat melalui citra wajah menggunakan metode facial landmark.
- Nisak, K., Sutomo, A. D., et al. (2021). Pengembangan sistem deteksi kantuk menggunakan pengklasifikasi random forest pada sinyal elektrokardiogram. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, 10(3), 265–271.

- Putilov, A. A., & Donskaya, O. G. (2013). Construction and validation of the eeg analogues of the karolinska sleepiness scale based on the karolinska drowsiness test. *Clinical Neurophysiology*, 124(7), 1346–1352.
- Raju, V. G., Lakshmi, K. P., Jain, V. M., Kalidindi, A., & Padma, V. (2020). Study the influence of normalization/transformation process on the accuracy of supervised classification. *2020 Third International Conference on Smart Systems and Inventive Technology (IC-SSIT)*, 729–735.
- Rogalska, A., Rynkiewicz, F., Daszuta, M., Guzek, K., & Napieralski, P. (2019). Blinking extraction in eye gaze system for stereoscopy movies. *Open Physics*, 17, 512–518. <https://doi.org/10.1515/phys-2019-0053>
- Roy, A., & Chakraborty, S. (2023). Support vector machine in structural reliability analysis: A review. *Reliability Engineering & System Safety*, 109126.
- Sanjaya, J., & Ayub, M. (2020). Augmentasi data pengenalan citra mobil menggunakan pendekatan random crop, rotate, dan mixup. *Jurnal Teknik Informatika dan Sistem Informasi*, 6(2).
- Sri Mounika, T., Phanindra, P., Sai Charan, N., Kranthi Kumar Reddy, Y., & Govindu, S. (2022). Driver drowsiness detection using eye aspect ratio (ear), mouth aspect ratio (mar), and driver distraction using head pose estimation. *ICT Systems and Sustainability: Proceedings of ICT4SD 2021, Volume 1*, 619–627.
- Valendion, V. (2022). Deteksi kantuk berdasarkan pola bukaan mulut menggunakan metode 1d cnn.
- Vicente, J., Laguna, P., Bartra, A., & Bailón, R. (2016). Drowsiness detection using heart rate variability. *Medical & biological engineering & computing*, 54, 927–937.
- Viola, P., & Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2), 137–154.
- Wilkinson, V. E., Jackson, M. L., Westlake, J., Stevens, B., Barnes, M., Swann, P., Rajaratnam, S. M., & Howard, M. E. (2013). The accuracy of eyelid movement parameters for drowsiness detection. *Journal of clinical sleep medicine*, 9(12), 1315–1324.

# LAMPIRAN

## Lampiran A : Lampiran Program Preprocessing Data

### A.1 Fungsi Menghitung Nilai EAR

```
1 def calculate_EAR(eye):
2     A = dist.euclidean(eye[1], eye[5])
3     B = dist.euclidean(eye[2], eye[4])
4     C = dist.euclidean(eye[0], eye[3])
5
6     eye_aspect_ratio = (A + B) / (2.0 * C)
7
8     return eye_aspect_ratio
```

### A.2 Fungsi Menghitung Nilai MAR

```
1 V = dist.euclidean(mouth[1], mouth[7])
2 W = dist.euclidean(mouth[2], mouth[6])
3 X = dist.euclidean(mouth[3], mouth[5])
4 Y = (V+W+X)/ 3.0
5
6 Z = dist.euclidean(mouth[0], mouth[4])
7
8 mouth_aspect_ratio = Y / Z
9
10 if mouth_aspect_ratio < 0.001:
11     mouth_aspect_ratio = 0.001
12 return mouth_aspect_ratio
```

### A.3 Program Interpolasi Data

```
1 for i in range (len(filenames)): #terahir index
2     data = pd.read_csv((inputNaN + filenames[i]), index_col=0, sep=',')
3     print(filenames[i])
4     print(data.isnull().sum())
5     for col in data.columns:
6         # menghitung ear
7         if col == 'ear_x' :
8             left_ear = calc_ear(data[data.columns[1]], data[data.columns[2]], ←
9                     data[data.columns[3]], data[data.columns[4]], ←
10                    data[data.columns[5]], data[data.columns[6]], ←
11                    data[data.columns[7]], data[data.columns←
12                    [8]], ←
13                    data[data.columns[9]], data[data.columns[10]], ←
14                    data[data.columns[11]], data[data.columns←
15                    [12]])
16             right_ear = calc_ear(data[data.columns[13]], data[data.columns←
17                     [14]], data[data.columns[15]], data[data.columns[16]], ←
18                     data[data.columns[17]], data[data.columns[18]], ←
19                     data[data.columns[19]], data[data.columns←
20                     [20]], ←
21                     data[data.columns[21]], data[data.columns[22]], ←
22                     data[data.columns[23]], data[data.columns←
23                     [24]])
24             value = pd.Series((left_ear+right_ear)/2)
```

```

15     # menghitung mar
16     elif col == 'mar_x':
17         mar = calc_mar(data[data.columns[26]], data[data.columns[27]], ←
18             data[data.columns[28]], data[data.columns[29]], ←
19                 data[data.columns[30]], data[data.columns[31]], ←
20                     data[data.columns[32]], data[data.columns←
21                         [33]], ←
22                         data[data.columns[34]], data[data.columns[35]], ←
23                             data[data.columns[36]], data[data.columns←
24                                 [37]], ←
25                         data[data.columns[38]], data[data.columns[39]], ←
26                             data[data.columns[40]], data[data.columns←
27                                 [41]])
28         value = pd.Series(mar)
29     else:
30         value = data[col].interpolate()
31         value = round(value,0)
32     data[col].fillna(value=value, inplace=True)
33     filenames[i] = filenames[i].split(".")[0]
34     data.to_csv('/mydrive/Output/' + filenames[i] + '.csv')

```

---

#### A.4 Fungsi Pemotongan Data

```

1 def begin_dat(data_num, data):
2     slice_dat = data.iloc[:data_num]
3     return slice_dat
4
5 def last_dat(data_num, data):
6     slice_dat = data.iloc[-data_num: ]
7     return slice_dat
8
9 def mid_dat(data_num, data):
10    len_dat = len(data.iloc[:,0])
11    mid = (len_dat - data_num) / 2
12    slice_dat = data[ceil(mid):-floor(mid)]
13    data.reset_index()
14
15    return slice_dat

```

---

#### A.5 Program Labeling Data

```

1 for i in range (len(filenames)):
2     data = pd.read_csv((output + filenames[i]), index_col=0, sep=',')
3     data.reset_index(drop = True, inplace = True)
4     df = data[['ear_x', 'mar_x']]
5     slice_dat = mid_dat(16200, df)
6     slice_dat.reset_index(drop = True, inplace = True)
7     nama_file = filenames[i]
8     label = label_name(nama_file)
9     slice_dat['label'] = label
10    print(slice_dat)
11    filenames[i] = filenames[i].split(".")[0]
12    slice_dat.to_csv('/mydrive/MarEar/' + filenames[i] + '.csv', index=←
13                      False)

```

---

## A.6 Program Normalisasi Data

---

```
1 from sklearn.preprocessing import StandardScaler
2
3 scaler = StandardScaler()
4 arr = scaler.fit_transform(data[['ear_x', 'mar_x']])
5 scaled_data = pd.DataFrame(arr, columns=['ear_x', 'mar_x'])
6 scaled_data
```

---

## A.7 Program Pembagian Data Training dan Data Validasi

---

```
1 def train_test_split(label, ratio):
2     split_point = int(len(data[data.label == label])*ratio)
3     return(data[data.label==label].iloc[:split_point,:], data[data.label←
4             ==label].iloc[split_point:, :])
5
6     split_ratio = (101/126)
7     train_data = pd.DataFrame([])
8     test_data = pd.DataFrame([])
9     total_label = 3
10    for i in range(total_label):
11        (train, test) = train_test_split(i+1, split_ratio)
12        train_data = pd.concat([train_data, train])
13        test_data = pd.concat([test_data, test])
```

---

## A.8 Program Sliding Window

---

```
1 WINDOW_LEN = 16200
2 #step lebih kecil harus overlap dengan window_len
3 STEP = 16199
4 N_FEATURE = 2
5
6 def generate_sequence(x, y, window_len, step):
7     segments = []
8     labels = []
9     for i in range(0, len(x)-window_len, step):
10         ear = x['ear_x'].values[i:i+window_len]
11         mar = x['mar_x'].values[i:i+window_len]
12         label = stats.mode(y['label'][i:i+window_len])[0][0]
13         segments.append([ear, mar])
14         labels.append(label)
15
16     return segments, labels
```

---

## Lampiran B : Lampiran Program Training Data B.1 Fungsi Callback

---

```
1 # buat callbacks
2 my_callbacks = [
3     tf.keras.callbacks.ReduceLROnPlateau(factor=0.5, patience=5, ),
4     tf.keras.callbacks.ModelCheckpoint(filepath='/mydrive/TA/←
5         IndRNNaug_best.h5', save_best_only=True)]
```

---

## B.2 Program Model IndRNN

---

```
1 def create_model():
2     rnn2= Sequential()
3     rnn2.add(Bidirectional(IndRNN(126, return_sequences=True, dropout←
4             =0.0, recurrent_dropout=0.0, kernel_initializer='orthogonal', ←
5                 kernel_regularizer=l2(L2), recurrent_regularizer=l2(L2),
```

```

4         bias_regularizer=l2(L2), name="LSTM_1"), input_shape=(←
5             WINDOW_LEN, N_FEATURE)))
6 rnn2.add(Dropout(0.3))
7 rnn2.add(Flatten(name='Flatten'))
8 rnn2.add(Dense(64, activation ='relu', name='Dense_3'))
9 rnn2.add(Dropout(0.2))
10 rnn2.add(Dense(32, activation ='relu', name='Dense_4'))
11 rnn2.add(Dropout(0.1))
12 rnn2.add(Dense(3, activation='softmax', kernel_regularizer=l2(L2), ←
13     bias_regularizer=l2(L2)))
14 rnn2.compile(optimizer=optimizers.Adam(0.000025), metrics=['accuracy'←
15     ], loss='categorical_crossentropy')
16 return rnn2
17
18 model = create_model()

```

---

### B.3 Program Model LSTM

```

1 lstm = Sequential()
2 lstm.add(Bidirectional(LSTM(126, return_sequences=True, ←
3     kernel_initializer='orthogonal', kernel_regularizer=l2(L2), ←
4     recurrent_regularizer=l2(L2),
5         bias_regularizer=l2(L2), name="LSTM_1"), input_shape=(←
6             WINDOW_LEN, N_FEATURE)))
7 lstm.add(Dropout(0.3))
8 lstm.add(Flatten(name='Flatten'))
9 lstm.add(Dense(64, activation ='relu', name='Dense_3'))
10 lstm.add(Dropout(0.2))
11 lstm.add(Dense(32, activation ='relu', name='Dense_4'))
12 lstm.add(Dropout(0.2))
13 lstm.add(Dense(3, activation='softmax', kernel_regularizer=l2(L2), ←
14     bias_regularizer=l2(L2)))
15 lstm.compile(optimizer=optimizers.Adam(0.000025), metrics=['accuracy'←
16     ], loss='categorical_crossentropy')

```

---

### B.4 Program Model SVM

```

1 svm_model = svm.SVC(kernel='rbf')
2
3 svm_model.fit(d2_train_dataset_x, ytrain)

```

---

### Lampiran C : Lampiran Program Evaluasi dan Testing Data

```

1 import seaborn as sns
2 from sklearn import metrics
3
4 # Recreate the exact same model, including its weights and the ←
5     optimizer
6 new_model = tf.keras.models.load_model('/mydrive/TA/IndRNNaug_best.h5←
7     ', custom_objects={'IndRNN':IndRNN})
8
9 # Show the model architecture
10 new_model.summary()
11
12 y_pred_ohe = new_model.predict(x_test) #menghasilkan 3 kelas
13 y_pred_ohe

```

---

```
13     y_pred_labels = np.argmax(y_pred_ohe, axis=1)
14     y_pred_labels
15
16     y_true_labels = np.argmax(y_test, axis=1)
17
18     confusion_matrix = metrics.confusion_matrix(y_true= y_true_labels, ←
19             y_pred= y_pred_labels)
20
21     plt.figure(figsize=(8,6))
22     sns.set(style='whitegrid')
23     sns.heatmap(confusion_matrix, xticklabels=['1', '2', '3'], ←
24             yticklabels=['1', '2', '3'], annot=True, fmt='d')
25     plt.title("confusion Matrix")
26     plt.ylabel("True Label")
27     plt.xlabel("predicted Label")
28     plt.show()
29
30     from sklearn.metrics import classification_report
31     print(classification_report(y_true_labels, y_pred_labels))
```

---

*[Halaman ini sengaja dikosongkan]*

## BIOGRAFI PENULIS



Adritia Alfiana Merdila, lahir di Gresik pada 2 November 2000. Penulis telah menempuh beberapa jenjang Pendidikan formal di beberapa sekolah yaitu: SMPN 2 Sidoarjo (2013-2016), dan SMAN 2 Sidoarjo (2016-2019). Penulis melanjutkan Pendidikan sarjana di Departemen Teknik Komputer Fakultas Teknologi Elektro dan Informatika Cerdas (FTEIC) Institut Teknologi Sepuluh Nopember (ITS) pada tahun 2019 yang terdaftar sebagai mahasiswa dengan nomor mahasiswa 07211940000017. Selama menjadi mahasiswa, penulis aktif mengikuti kegiatan kepanitiaan diantaranya Evolve, MAGE 6, dan MAGE 7. Penulis juga aktif mengikuti beberapa organisasi seperti menjadi Bendahara di Himpunan Mahasiswa Teknik Komputer (HIMATEKKOM). Penulis tertarik dengan bidang machine learning dan pengembangan web. Untuk mendapatkan gelar S.T (Sarjana Teknik), penulis mengambil topik penelitian tugas akhir deep learning khususnya yang berkaitan dengan IndRNN. Untuk kepentingan penelitian, pembaca yang memiliki kritik, saran, atau pertanyaan mengenai tugas akhir ini dapat menghubungi penulis melalui email [adritiamerdila.19072@mhs.its.ac.id](mailto:adritiamerdila.19072@mhs.its.ac.id).

*[Halaman ini sengaja dikosongkan]*