Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

Optimalisasi Arsitektur LSTM dengan Pendekatan Bidirectional untuk Deteksi Kantuk Pengemudi Berbasis Fitur Wajah

Andhika Rhaifahrizal Hartono, Muhammad Naufal*, Farrikh Alzami

Fakultas Ilmu Komputer, Teknik Informatika, Universitas Dian Nuswantoro, Semarang, Indonesia
Email: [1]111202113231@mhs.dinus.ac.id, [2,*]m.naufal@dsn.dinus.ac.id, [3]alzami@dsn.dinus.ac.id
Email Penulis Korespondensi: m.naufal@dsn.dinus.ac.id
Submitted: **12/08/2025**; Accepted: **09/09/2025**; Published**: 10/09/2025**

Abstrak−Kecelakaan lalu lintas akibat kelelahan dan kantuk pengemudi merupakan isu keselamatan yang serius di banyak negara, termasuk Indonesia. Berbagai sistem deteksi kantuk berbais citra telah dikembangkan, namun beberapa masih bergantung pada analisis frame tunggal dan kurang mampu menangkap konteks temporal secara menyeluruh. Untuk mengatasi masalah ini, diperlukan sistem yang mampu mendeteksi tanda-tanda kantuk secara akurat dan real-time. Penelitian ini bertujuan mengevaluasi dan membandingkan kinerja algoritma Long Short-Term Memory (LSTM) dan Bidirectional LSTM (BiLSTM) untuk sistem deteksi kantuk berbasis fitur wajah. Dataset yang digunakan adalah YawDD, berisi video perilaku menguap dan kondisi netral pengemudi. Setiap video diekstrak menjadi frame dan dianalisis menggunakan MediaPipe untuk mendapatkan landmark wajah. Dua fitur utama yang digunakan adalah Eye Aspect Ratio (EAR) dan Mouth Opening Ratio (MOR). Dikarenakan data tidak imbang maka dilakukan teknik SMOTE pada kelas minoritas pada data train. Model LSTM dan BiLSTM dibandingkan dengan konfigurasi arsitektur yang serupa. Hasil menunjukkan bahwa BiLSTM unggul dengan akurasi 94,74% dan F1-score 94,82%, dibandingkan LSTM yang mencapai akurasi 92,98% dan F1-score 93,22%. Temuan ini menunjukkan bahwa pemrosesan data sekuensial dua arah pada BiLSTM lebih efektif dalam mengenali pola temporal dari gejala kantuk. Penelitian ini memberikan kontribusi bagi pengembangan sistem deteksi kantuk berbasis citra komputer yang akurat dan efisien.

Kata Kunci: Deteksi Kantuk; LSTM; BiLSTM; Eye Aspect Ratio; Mouth Opening Ratio

Abstract−Traffic accidents caused by driver fatigue and drowsiness remain a serious safety concern in many countries, including Indonesia. Various image-based drowsiness detection systems have been developed, yet many still rely on single-frame analysis and lack the ability to capture complete temporal context. To address this issue, a system capable of accurately and real-time detecting signs of drowsiness is required. This study aims to evaluate and compare the performance of Long Short-Term Memory (LSTM) and Bidirectional LSTM (BiLSTM) algorithms for a facial-feature-based drowsiness detection system. The dataset used is YawDD, which consists of videos of drivers yawning and in neutral conditions. Each video was decomposed into frames and analyzed using MediaPipe to extract facial landmarks. Two main features, Eye Aspect Ratio (EAR) and Mouth Opening Ratio (MOR), were utilized. Due to class imbalance, the SMOTE technique was applied to the minority class in the training data. Both LSTM and BiLSTM models were compared under similar architecture configurations. The results show that BiLSTM outperformed LSTM with an accuracy of 94,74% and an F1- score 94,82%, compared to 92,98% accuracy and 93,22% F1-score achieved by LSTM. These findings demonstrate that bidirectional sequential processing in BiLSTM is more effective in capturing the temporal patterns of drowsiness symptoms. This study contributes to the development of accurate and efficient computer vision-based drowsiness detection systems.

Keywords: Drowsiness Detection; LSTM; BiLSTM; Eye Aspect Ratio; Mouth Opening Ratio

1. PENDAHULUAN

Kecelakaan lalu lintas menjadi salah satu isu penting yang dapat memengaruhi keselamatan publik, menimbulkan korban jiwa, serta kerugian material yang signifikan baik bagi korban maupun pihak yang terlibat dalam kecelakaan. Jumlah angka kecelakaan lalu lintas di Indonesia terus meningkat seiring berjalannya waktu. Berdasarkan data dari Badan Pusat Statistik (BPS) Indonesia pada tahun 2022, tercatat peningkatan sebesar 35.613 kasus kecelakaan dibandingkan dengan tahun sebelumnya[1]. Beberapa faktor yang dapat meningkatkan risiko kecelakaan adalah kelelahan dan kantuk pengemudi. Kelelahan dapat menurunkan konsentrasi pengemudi, memperlambat waktu reaksi, serta meningkatkan kemungkinan pengemudi melakukan kesalahan yang berujung pada kecelakaan[2].

Beberapa tahun terakhir, sejumlah metode telah dikembangkan untuk mendeteksi kantuk pengemudi, terutama dengan memanfaatkan teknologi computer berbasis visi. Penelitian Vaman memanfaatkan Histogram of Oriented Gradients (HOG) dalam mengklasifikasikan ekspresi wajah [3]. Tinaliah menggunakan CNN untuk melakukan klasifikasi pada ekspresi wajah manusia [4]. Kedua metode tersebut memiliki satu kesamaan yaitu kemampuannya untuk mendeteksi objek pada gambar tunggal. Karakteristik yang bersifat frame-based ini memunculkan keterbatasan dalam menangkap informasi yang bersifat dinamis. Dalam kehidupan nyata, gejala kantuk seperti menguap baru dapat diamati melalui perubahan berurutan dalam rentang waktu tertentu. Untuk mengatasi keterbatasan pendekatan tersebut, penelitian mengusulkan metode berbasis temporal dengan mendapatkan informasi berurutan dari sejumlah sekuens. Long Short-Term Memory (LSTM) merupakan algoritma berbasis sequensial sebagai solusi dengan memanfaatkan mekanisme gerbang (gates) yang memungkinkan untuk menghilangkan permasalahan vanishing gradient pada RNN.

Penelitian yang dilaksanakan oleh Chen[5] berhasil menujukkan bahwa LSTM dapat digunakan untuk mendeteksi kantuk berbasis fitur wajah pada dataset sekuensial. Dengan hasil akurasi mencapai 88%, meskipun hasil ini masih terbatas oleh arsitektur yang hanya memproses informasi secara satu arah. Sebagai pengembangan lebih lanjut, Bidirectional LSTM (BiLSTM) dikembangkan dengan kemampuan memproses data dari dua arah sekaligus.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

Dengan adanya mekanisme ini, BiLSTSM dapat menangkap pola temporal yang lebih lengkap karena memanfaatkan informasi dari masa lalu dan masa depan secara bersamaan[6]. Zhao berhasil mengimplementasikan BiLSTM dalam penelitiannya untuk mengklasifikasikan dan mendeskripsikan video pola pergerakan atlet dengan tingkat akurasi sebesar 95,03%, lebih tinggi dibandingkan LSTM konvensional[7]. Temuan ini menegaskan bahwa BiLSTM relevan digunakan dalam sekuensial dan dapat dimanfaatkan pada tugas deteksi kantuk, dimana gejala kantuk tidak hanya bergantung pada kondisi satu frame. Namun, mempertimbangkan rangkaian perubahan sebelum dan sesudah suatu peristiwa.

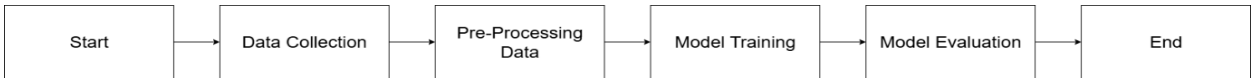
Selain pemilihan arsitektur model, penentuan fitur juga menjadi hal penting dalam sistem deteksi kantuk. Berbagai pendekatan yang ada dapat dikategorikan menjadi tiga kelompok, yaitu berbasis sinyal biologis, perilaku kendaraan, dan karakteristik visual pengemudi[8]. Penelitian memfokuskan pada fitur wajah berbasis landmark wajah Eye Aspect Ratio (EAR) dan Mouth Opening Ratio (MOR) yang digunakan karena dapat memberikan indikasi yang jelas terhadap kondisi kantuk pada pengemudi, seperti kelopak mata yang menutup saat mengantuk dan mulutyang terbuka lebar saat kondisi menguap.

Oleh karena itu, penelitian ini bertujuan untuk mengevaluasi keunggulan pemrosesan konteks dua arah dengan membandingkan kinerja LSTM dan Bi-LSTM dalam mendeteksi kantuk pengemudi berdasarkan data sekuensial. Dengan pendekatan ini, penelitian diharapkan dapat memberikan kontribusi dalam pengembangan sistem deteksi kantuk yang lebih akurat dan mendukung peningkatan keselamatan dan keamanan berkendara.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Gambar 1 menunjukkan tahapan-tahapan yang dilakukan dalam penelitian ini, yang dimulai dengan mengumpulkan dataset pengemudi kendaraan yang mengantuk, kemudian dilanjutkan dengan tahapan pre-processing data. Lalu data yang sudah diproses dibagi menjadi tiga yaitu data train, evaluation, dan test. Ketiga data tersebut akan digunakan pada tahapan pelatihan model dan evaluasi model.



Gambar 1. Tahapan Penelitian

Secara garis besar, Gambar 1 memberikan gambaran alur penelitian mulai dari proses ekstraksi fitur pada dataset hingga tahapan evaluasi model. Setiap tahapan yang ditampilkan pada diagram diatas dijelaskan secara lebih rinci pada subbab 2.2 hingga 2.6.

2.2 Data Collection

Penelitian ini menggunakan dataset video YawDD (Yawning Detection Dataset). Dataset ini merupakan kumpulan rekaman di dalam mobil dengan berbagai karakteristik pengemudi. Karakteristik tersebut mencakup aspek fisik seperti jenis kelamin, etnis, dan gaya berpakaian, serta perilaku pengemudi seperti kondisi netral (terdiam) dan menguap[9]. Video yang digunakan yaitu video dengan label normal dan *yawning* (menguap) dengan total 181 video dengan durasi 8 hingga 30 detik.

2.2.1 Ekstraksi Fitur dengan Mediapipe

Dataset video kemudian diekstaksi menggunakan mediapipe landmark menjadi data numerik melalui proses ekstraksi frame. Proses ini dilakukan pada setiap frame dengan menggunakan framework *MediaPipe* untuk mendeteksi 468 titik *landmark* pada wajah secara real-time[10]. Dengan memanfaatkan titik *landmark* ini, dapat diperoleh dua parameter utama, yaitu nilai titik landmark EAR dan MOR yang kemudian diproses menjadi data .csv.

Gambar 2. Ilustrasi Facemesh MediaPipe

Gambar 2 memperlihatkan ilustrasi titik landmark pada wajah yang dihasilkan oleh MediaPipe Facemesh dengan total 468 titik koordinat. Titik-titik ini berperan penting sebagai dasar perhitungan fitur EAR dan MOR pada penelitian ini.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

2.2.2 Eye Aspect Ratio (EAR)

Eye Aspect Ratio (EAR) digunakan untuk mengukur derajat keterbukaan mata dengan batas nilai yang sudah ditentukan. Derajat keterbukaan mata dihitung dari jarak enam titik *landmark* secara vertikal pada titik kelopak mata atas dan titik kelopak mata bawah [11]. Secara horizontal pada lebar kelopak mata yang sudah ditetapkan menggunakan *MediaPipe* pada masing-masing mata. Untuk mendapatkan nilai EAR dapat didefinisikan seperti (1).

$$EAR = \frac{\|P_2 - P_6\| + \|P_3 - P_5\|}{2\|P_1 - P_4\|} \quad (1)$$

Nilai EAR akan menurun ketika kondisi mata tertutup dan akan tetap rendah selama kondisi pengendara terindikasi kantuk. Dalam penelitian ini batasan nilai EAR yang ditetapkan adalah 0.2. Berdasarkan penelitian sebelumnya[12], jika nilai EAR berada dibawah *threshold* tersebut secara berutur-turut maka kondisi tersebut diindikasikan sebagai mata tertutup atau kondisi kantuk. Pada penelitian ini, tititk *landmark* pada mata yang digunakan adalah sebagai berikut: [33, 160, 158, 133, 153, 144].

2.2.3 Mouth Opening Ratio (MOR)

Mouth Opening Ratio (MOR) merupakan parameter yang digunkan untuk mendeteksi aktivitas bukaan mulut secara signifikan seperti saat pengemudi sedang menguap. Nilai MOR dapat dihitung berdasarkan jarak titik landmark vertikal antara bibir atas dan bawah terhadap lebar mulut secara horizontal [8]. Untuk mendapatkan nilai MOR dapat didefinisikan seperti (2).

$$MOR = \frac{\|P_2 - P_6\| + \|P_3 - P_5\|}{2\|P_1 - P_4\|} \quad (2)$$

Jika nilai MOR melebihi ambang batas yang sudah ditetapkan maka kondisi tersebut akan diidentifikasikan sebagai aktivitas menguap atau kondisi kantuk pada mengemudi. Titik *landmark* yang digunakan untuk mendapatkan nilai MOR pada penelitian ini didefinisikan sebagai berikut: [78, 81, 402, 308, 311, 178].

2.2.4 Sliding Window

Nilai EAR dan MOR yang berhasil diekstrak dikumpulkan menjadi data berbentuk sekuensial yang sesuai untuk training model. Teknik *Sliding-Window* digunakan untuk mengelompokkan nilai EAR dan MOR pada setiap sekuens. Teknik *Sliding-Window* merupakan pendekatan untuk membagi data sekuens menjadi beberapa potongan dengan ukuran jendela (*window size*) yang tetap dan berpindah secara bertahap sesuai dengan ukuran langkah (step size) yang ditentukan[13]. Dalam penelitian ini, digunakan *window size* sebesar 90 frame dan *step size* sebesar 30 frame. Dengan demikian, setiap sekuens terdiri dari 90 data EAR dan MOR yang mewakili 3 detik durasi video.

2.3 Pre-Processing Data

Pada tahapan ini beberapa langkah dilakukan untuk memastikan data siap digunakan sebelum proses training model. Pertama, diterapkan metode *moving average* dengan ukuran *window size* sebesar 5 pada setiap fitur dalam satu sekuens. Tujuan penerapan *moving average* adalah untuk mereduksi noise dan fluktuasi pada data, sehingga pola perubahan data menjadi lebih halus dan mudah dipelajari oleh model[14]. Perhitungan *moving average* didapatkan melalui rumus berikut (3).

$$X'_t = \frac{X_t + X_{t-1} + X_{t-2} + \ldots + X_{t-(n-1)}}{n} \quad (3)$$

Pada persamaan tersebut, X_t merepresentasikan nilai data asli pada waktu ke-t, sedangkan X'_t adalah nilai haril moving average pada waktu ke-t. parameter n menujukkan ukuran jendela (window size) yang digunakan dalam proses pemerataan data. Dengan menggunakan pendekatan ini, pola tren data dapat menjadi lebih halus sehingga meminimalisir noise sebelum masuk ke tahap pelatihan model.

Setelah proses *smoothing* data dilakukan, dataset dibagi menjadi tiga subset dengan rasio 70% untuk data train, 15% untuk data validation, dan 15% untuk data test. Data train digunakan untuk mempelajari pola perubahan data pada saat pelatihan model, sedangkan data validation digunakan untuk mengevaluasi kinerja model saat proses training dan sebagai acuan performa model yang dikembangkan[15]. Data test digunakan untuk mengukur performa model secara menyeluruh setelah proses training selesai. Selanjutnya, dilakukan proses normalisasi pada seluruh data menggunakan metode *Min-Max Scaler* agar setiap fitur memiliki bobot dan rentang nilai yang sama antara 0 dan 1[16]. Rumus perhitungan *Min-Max Scaler* dapat dilihat pada (4).

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (4)$$

Pada persamaan tersebut, X_{sc} merepresentasikan nilai data hasil normalisasi, X merupakan nilai data asli sebelum dilakukan normalisasi, sementara X_{min} dan X_{max} menunjukkan nilai minimum dan maksimum dari suatu atribut. Normalisasi ini memastikan semua fitur berada pada rentang 0 hingga 1 sehingga dapat menghindari bias akibat perbedaan skala antar fitur.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

2.4 Synthetic Minority Over-Sampling Technique (SMOTE)

Setelah melalui tahap pre-processing, dilakukan penanganan terhadap ketidakseimbangan kelas pada data train dengan penerapan SMOTE (*Synthetic Minority Over-sampling Technique*). SMOTE bekerja dengan cara menghasilkan sampel sintetis baru dari kelas minoritas melalui interpolasi antar data yang sudah ada [17]. Dengan demikian, jumlah sampel pada kelas minoritas bertambah dan distribusi kelas menjadi lebih seimbang. Pada penelitian ini, SMOTE hanya diterapkan hanya pada data train untuk mencegah adanya *data leakage* pada data validation maupun data test saat proses training model. Rumus perhitungan SMOTE didefinisikan seperti (5).

$$X_{new} = X_i + (X_j - X_i) \times \delta \tag{5}$$

Pada persamaan tersebut X_{new} adalah sampel sintetis baru, X_i merupakan data asli yang dipilih dari kelas minoritas, X_j adalah tetangga terdekat dari X_i, dan δ adalah bilangan acak antara 0 dan 1. Dalam konteks data sekuensial, setiap sampel data pada penelitian ini berupa rangkaian vector fitur dalam jendela waktu 90 frame per sekuens. Oleh karena itu, penerapan SMOTE dilakukan dengan memperlakukan satu sekuens penuh sebagai data berdimensi tinggi. Proses interpolasi kemudian diterapkan antara sekuens kelas minoritas dengan sekuens tetangga terdekatnya pada ruang fitur. Sehingga distribusi data antar kelas menjadi lebih seimbang tanpa menghilangkan informasi temporal dari data asli.

2.5 Perancangan Model

Model LSTM dan BiLSTM yang digunakan, dirancang dan dikonfigurasi pada setiap *layer* model yang digunakan. *Long Short-Term Memory* (LSTM) adalah salah satu arsitektur *RNN* yang sudah dikembangkan lebih lanjut dengan kelebihan dapat memecahkan permasalahan yang terdapat pada algoritma RNN seperti permasalahan *vanishing gradient* [18]. LSTM tediri dari tiga struktur gerbang yaitu *input gate*, *output gate*, dan *forget gate* yang masing-masing berfungsi untuk menentukan informasi apa saja yang perlu ditambahkan, dijadikan output, dan dibuang dari memori[19]. Persamaan yang digunakan dalam ketiga proses tersebut dapat diformulakan sebagai berikut.

$$f_t = \sigma\big(W_f \cdot [h_{t-1}, x_t] + b_f\big) \tag{6}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_i] + b_i) \tag{7}$$

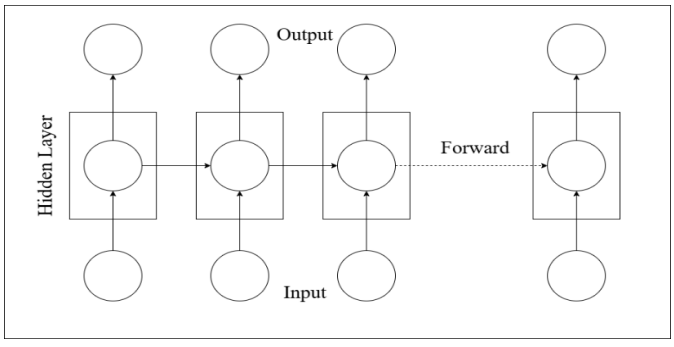
$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{8}$$

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t \tag{9}$$

$$O_t = \sigma(W_o \cdot [h_{t-1}, x_i] + b_o) \tag{10}$$

$$h_t = O_t \times \tanh C_t \tag{11}$$

Berdasarkan formula diatas, f_t merepresentasikan forget gate yang berfungsi untuk mengatur informasi yang perlu dipertahankan atau dibuang dari memori sebelumnya. Input gate yang dilambangkan dengan i_t menentukan informasi baru yang perlu untuk ditambahkan dalam memori. $\tilde{C}_t$ merupakan kandidat nilai baru yang akan dimasukkan dalam memori dengan fungsi aktivasi $tanh$. Persamaan C_t merepresentasikan memori terkini yang menggabungkan informasi lama dan informasi baru. Output gate yang direpresentasikan dengan persamaan O_t, menentukan besarnya informasi dari memori yang akan dikeluarkan. Persamaan h_t menjadi hidden state pada pada waktu ke- t yang merepresentasikan keluaran utama LSTM.



Gambar 2. Ilustrasi Arsitektur LSTM

Gambar 2 memperlihatkan representasi visual arsitektur LSTM. Ilustrasi ini menekankan bagaimana aliran data dikelola melalui tiga gerbang utama untuk mengatur informasi yang masuk sebagai input, informasi yang dipertahankan dalam memori, maupun dikeluarkan sebagai output.

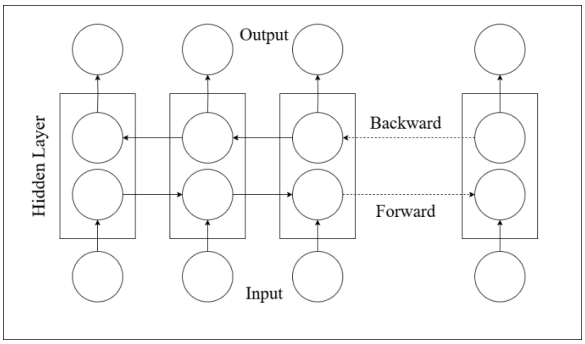
Bidirectional Long Short-Term Memory (BiLSTM) merupakan perkembangan lanjut dari model LSTM. Dengan mengimplementasikan cara kerja yang sama dengan *Bidirectional RNN*, BiLSTM dapat mengolah input data sekuensial tidak hanya dari arah maju (*forward*) namun juga dari arah mundur (*backward*) [20]. Hal ini

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

memungkinkan BiLSTM untuk mempelajari pola dan informasi masa lalu dan masa depan dalam sebuah sekuens dengan lebih baik dibandingkan dengan LSTM biasa[21]. Pada arsitektur BiLSTM persamaan yang digunakan dapat dirumuskan sebagai berikut.

$$h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}] \tag{12}$$

Pada persamaan diatas, nilai keluaran BiLSTM pada waktu ke- t dilambangkan dengan h_t. Komponen $\overrightarrow{h_t}$ menunjukkan hasil perhitungan LSTM pada arah maju (forward), sedangkan $\overleftarrow{h_t}$ menunjukkan hasil perhitungan LSTM pada arah mundur (backward).



Gambar 3. Ilustrasi Arsitektur BiLSTM

Gambar 3 memperlihatkan ilustrasi arsitektur BiLSTM yang terdiri dari dua lapisan LSTM yang memproses data sekuensial secara paralel dalam arah yang berlawanan. Hasil pemrosesan dari kedua arah tersebut kemudian digabungkan pada setiap Langkah waktu, sehingga model dapat menangkap pola temporal secara lebih menyeluruh dibandingkan dengan LSTM satu arah.

Optimizer *Adam* (Adaptive Moment Estimation) merupakan algoritma optimisasi berbasis stochastic gradient descent yang dapat menyesuaikan diri dan banyak digunakan pada deep learning. Optimizer *Adam* digunakan karena penggunaannya yang kompatibel dengan berbagai arsitektur dan model, serta kemampuannya dalam menyesuaikan *learning rate* pada setiap parameter dengan efisien[22]. Proses perhitungan optimizer *Adam* dapat dituliskan melalui rumus berikut.

$$m_1 = \beta_1 \times m_1 + (1 - \beta_1) \times gradient \tag{13}$$

$$m_2 = \beta_2 \times m_2 + (1 - \beta_2) \times gradient \tag{14}$$

$$\hat{m}_1 = m_1 \div (1 - {\beta_1}^t) \tag{15}$$

$$\hat{m}_2 = m_2 \div (1 - {\beta_2}^t) \tag{16}$$

$$p = p - lr \times \hat{m}_1 \left(\sqrt{\hat{m}_2 + \varepsilon}\right) \tag{17}$$

Pada persemaan tersebut, m_1 dan m_2 mewakilkan momentum orde pertama dan orde kedua. Parameter β berfungsi untuk mengatur peluruhan momentum pada setiap iterasi. Nilai $\hat{m}_1$ dan $\hat{m}_2$ merupakan hasil koreksi bias. Symbol *gradient* menunjukkan gradien dari loss fungction terhadap parameter p. lr adalah learning rate, sedangkan ε ditambahkan untuk mencegah terjadinya pembagian dengan nol.

Binary Cross-Entropy (BCE) merupakan salah satu loss function yang digunakan pada klasifikasi biner untuk mengukur perbedaan antara probabilitas prediksi model dan label sebenarnya. Fungsi ini menghitung nilai loss secara independen untuk setiap komponen output[23]. Rumus umum untuk menghitungnilai loss dengan BCE dapat dituliskan sebagai berikut.

$$BCE = -\frac{1}{N}\sum_{i=1}^{N}[y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)] \tag{18}$$

2.6 Performance Measurement

Pengujian model akan dilakukan dengan data test yang sudah dibagi sebelumnya, yang dimana 15% dari dataset akan digunakan untuk pengujian klasifikasi dan untuk mendapatkan hasil akurasi dari keseluruhan dataset. Penelitian ini menggunakan classification report untuk mendapatkan nilai pada *confusion matrix*. Nilai ini mencatat jumlah keseluruhan *True Positives* (TP), *False Positves* (FN), *True Negatives* (TN), dan *False Negatives* (FN). Dengan menggunakan nilai yang didapatkan melalui *confusion matrix* tersebut, nilai *recall*, *precision*, dan *F1-score* pada *weighted average* bisa didapatkan untuk membandingkan performa LSTM dan BiLSTM.

$$Recall = \frac{TP}{TP+FN} \tag{19}$$

$$Precision = \frac{TP}{TP+FP} \tag{20}$$

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (21)$$

$$F1\ Score = \frac{2\times(Recall\times Precision)}{(Recall+Precision)} \quad (22)$$

Setiap metrik penilaian memiliki tujuan yang berbeda. *Recall* digunakan untuk mengukur kemampuan model dalam mengenali data positif secara menyeluruh, sedangkan *precision* menilai ketepatan model dalam mengklasifikasikan data positif dengan benar dibandingkan dengan seluruh prediksi positif yang dihasilkan. *Accuracy* menunjukkan proporsi prediksi yang benar dari keseluruhan kelas positif maupun negatif, sehingga *accuracy* dapat memberikan gambaran umum mengenai keakuratan model. *F1-score* merupakan perhitungan rata-rata dari *precision* dan *recall* yang memberikan gambaran evaluasi yang lebih seimbang pada performa model, khususnya saat jumlah data antar kelas tidak seimbang[24].

3. HASIL DAN PEMBAHASAN

3.1 Hasil Data Collection

Berdasarkan hasil ekstraksi fitur pada dataset *YawDD*[25] menggunakan *MediaPipe*, didapatkan total 3038 sekuens data dengan dua kelas, yaitu *normal* sebanyak 2368 sekuens dan *yawning* sebanyak 670 sekuens. Setiap sekuens terdiri atas 90 pasang nilai *Eye Aspect Ratio* (EAR) dan *Mouth Opening Ratio* (MOR) yang mewakili jumlah frame dalam satu sekuens, serta kolom label sebagai penanda kondisi pengemudi berdasarkan nilai EAR dan MOR yang tersebut. Nilai rata-rata EAR tercatat sebesar 0,23 dengan rentang 0,01 hingga 0,72. Sedangkan nilai rata-rata MOR tercatat sebesar 0,09 dengan rentang 0,00 hingga 1,19. Nilai EAR yang berada di bawah batas 0,2 mengindikasikan kondisi mata tertutup, sementara nilai MOR diatas batas 0,5 mengindikasikan kondisi mulut yang sednag menguap. Dengan demikian, kedua parameter ini dapat digunakan untk membedakan kondisi pengemudi antara keadaan *normal* dan keadaan mengantuk (*yawning*). Contoh hasil ekstraksi fitur EAR dan MOR ditunjukkan pada Tabel 1.

Tabel 1. Contoh Hasil Ekstraksi Fitur

EAR_1	MOR_1	EAR_2	MOR_2	...	EAR_90	MOR_90	Label
0.32	0.03	0.32	0.04		0.29	0.03	Normal
0.17	0.13	0.12	0.15		0.16	0.00	Yawning
0.25	0.00	0.23	0.00		0.23	0.00	Normal

Tabel 1 memperlihatkan contoh representasi data hasil ekstraksi fitur dalam bentuk sekuens yang digunakan sebagai input pada tahap pemodelan. Jumlah tersebut merepresentasikan 90 frame pada video berdurasi 3 detik. Pemilihan durasi didasarkan pada pertimbangan gejala kantuk saat mengemudi umumnya terjadi dalam rentang waktu 3 hingga 10 detik [26].

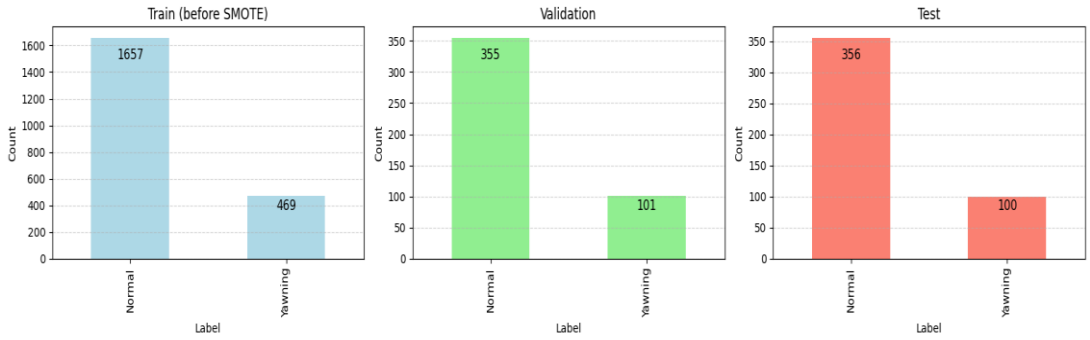
3.2 Hasil Pre-Processing Data

Dari data yang diperloleh dilakukan label encoding dimana normal berubah menjadi 0 dan kantuk menjadi 1. Kemudain hasil analisis awal terhadap dataset memperlihatkan bahwa jumlah sekuens pada kelas *Normal* lebih dominan dibandingkan kelas *Yawning*. Ketidakseimbangan ini berpotensi mengurangi kemampuan model dalam mengenali pola dari kelas minoritas, sehingga diperlukan tahap *pre-processing* untuk menyeimbangkan distribusi data. Salah satu langkah yang dilakukan adalah menerapkan metode *moving average* dengan *window size* sebesar 5 pada setiap sekuens. Penerapan metode ini terbukti mampu mereduksi fluktuasi pada nilai EAR dan MOR, sehingga pola perubahannya lebih stabil dan tidak ekstrim. Setelah itu, dilakukan normalisasi menggunakan metode *Min-Max Scaler* agar fitur berada pada rentang 0 hingga 1. Normalisasi memastikan skala antar fitur menjadi lebih seragam serta mempermudah proses training model dengan lebih stabil. Dari seluruh dataset yang didapat, memiliki sebaran *normal* 2368 sekuens dan *yawning* sebanyak 670 sekuens atau sebesar 1:3.53 yang menandakan bahwa persebaran data tersebut *imbalance*.

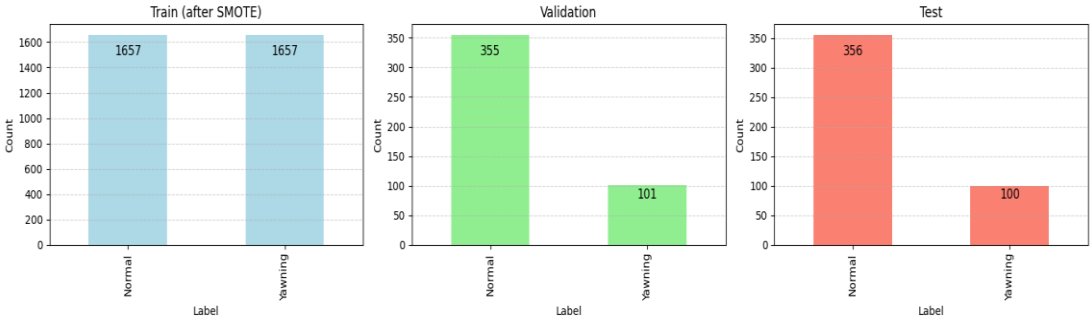
Selanjutnya, dataset dibagi menjadi tiga subset, yaitu train 70%, val 15%, dan 15% untuk data test dengan proporsi sesuai data asli. Dari hasil pembagian ini diperoleh 2126 sekuens untuk training, 456 sekuens untuk validation, dan 456 sekuens untuk test. Pada data train, jumlah sampel kelas *normal* adalah 1657 sekuens, dan kelas *yawning* sebanyak 469 sekuens. Sementara itu, pada data validation dan test masing-masing terdiri dari 355 sekuens kelas *normal* dan 101 sekuens kelas *yawning*.

Distribusi data ini memperlihatkan adanya ketidakseimbangan jumlah sampel antar kelas yang berpotensi menimbulkan bias pada model. Untuk mengatasi hal tersebut, penelitian ini menerapkan SMOTE pada data train dengan tujuan menyeimbangkan jumlah sekuens antara kelas mayoritas (*Normal*) dan kelas minoritas (*Yawning*). Penerapan SMOTE dibatasi hanya pada data train untuk mencegah terjadinya kebocoran data. Perbedaan distribusi data sebelum dan sesudah penerapan SMOTE dapat diamati pada Gambar 4 dan Gambar 5.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

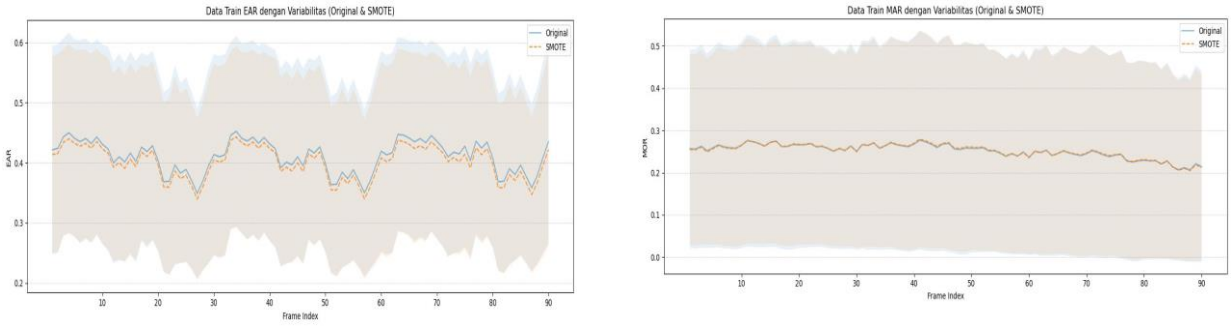


Gambar 4. Distribusi Data Asli



Gambar 5. Distribusi Data Setelah SMOTE

Gambar 4 memperlihatkan distribusi data sebelum penerapan SMOTE, dimana jumlah sekuens pada kelas *normal* jauh lebih besar dibandingkan kelas *yawning*. Sedangkan, Gambar 5 menunjukkan sebaran data setelah penerapan SMOTE pada data train dengan jumlah yang seimbang antar kelas. Dan untuk perbandingan data hasil SMOTE dengan data train yang asli divisualkan pada Gambar 6 berikut. Visualisasi tersebut menunjukan data sintetis yang dihasilkan sebarannya sejalan dengan sekuens data train.



(a) (b)

Gambar 6. Data Train Hasil SMOTE (a) EAR, (b) MOR

3.3 Hasil Perancangan Model

Pada tahapan ini kedua model, yaitu LSTM dan BiLSTM dirancang menggunakan konfigurasi parameter dan jumlah lapisan (*layer*) yang serupa dengan tujuan agar performa kedua model dapat dibandingkan secara objektif. Kedua model dilatih menggunakan optimizer *Adam* serta fungsi loss *Binary Cross Entropy*. Selama proses pelatihan, model dilatih dengan jumlah *epoch* sebanyak 100, *batch size* sebesar 32, serta *learning rate* sebesar 0,0001. Pemilihan parameter tersebut bertujuan agar model dapat memepalajari pola data sekuensial secara menyeluruh, sekaligus meminimalkan kemungkinan kegagalan pembaruan bobot selama proses training. Penggunaan *learning rate* yang kecil bertujuan untuk meningkatkan ketelitian pembelajaran model, sehingga proses konvergensi dapat berlangsung secara stabil tanpa melewati nilai minimum lokal[27].

3.3.1 LSTM

Model LSTM yang dirancang pada penelitian ini dibangun dengan arsitektur berlapis untuk mengolah data sekuensial searah maju. Data input dalam bentuk vector EAR dan MOR sepanjang 90 timestep diproses oleh lapisan LSTM pertama dengan 64 unit untuk mengekstraksi pola temporal awal, dan terdapat layer seperti batch normalization, dense dropout dan regulasi yang digunakan untuk yang lengkapnya model ditunjukkan pada Tabel 2.

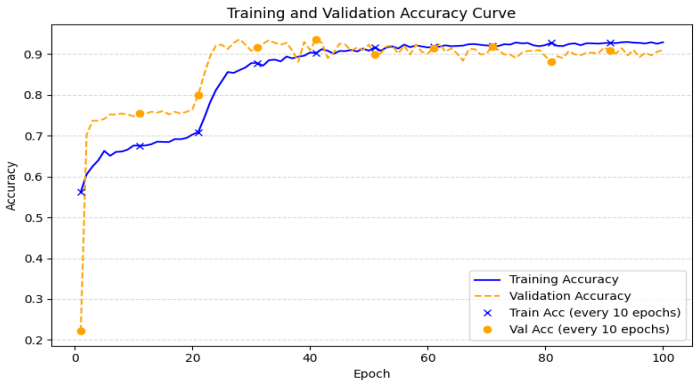
Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

Tabel 2. Arsitektur Model LSTM

Layer	Input	Output	Parameter
Input	(None, 90, 2)	(None, 90, 2)	
LSTM	(None, 90, 2)	(None, 90, 64)	17.152
Batch Normalization	(None, 90, 64)	(None, 90, 64)	256
LSTM	(None, 90, 64)	(None, 32)	12.416
Dropout	(None, 32)	(None, 32)	0
Dense	(None, 32)	(None, 32)	1.056
Dropout	(None, 32)	(None, 32)	0
Dense	(None, 32)	(None, 1)	33

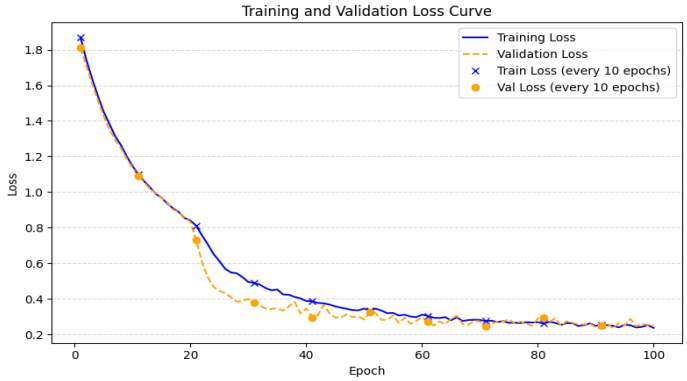
Detail konfigurasi pada Tabel 2 memperlihatkan bahwa lapisan LSTM pertama memiliki 17.152 parameter yang mencerminkan kompleksitas dalam menagnkap pola sekuensial awal. Batch Normalization digunakan untuk menstabilkan distribusi input antar layer. Lapisan LSTM kedua dengan 12.416 parameter berperan dalam menyaring informasi sehingga representasi yang dihasilkan lebih padat namun tetap bermakna. Lapisan Dense dengan activation function ReLU dan jumlah neuron sebanyak 32 unit, menambahkan 1.056 parameter untuk memproyeksikan hasil sebelumnya ke dimensi klasifikasi. Pada lapisan LSTM maupun Dense, L2 Regularization diterapkan sehingga bobot model yang besar dikenai penalty. Mekanisme ini dapat mencegah kompleksitas berlebih yang dapat menyebabkan overfitting pada model. Sementara penggunaan dua lapisan Dropout dengan masing-masing dropout rate sebesar 0,5 juga berfungsi untuk membantu mengurangi fluktuasi dan mencegah terjadinya overfitting pada model. Lapisan output Sigmoid dengan 1 unit neuron menghasilkan probabilitas prediksi untuk klasifikasi biner. Jumlah parameter trainable yang digunakan dalam proses training model LSTM ini yaitu 30.785. hal ini menunjukkan bahwa model cukup efisien namun tetap memiliki kapasitas yang memadai untuk mempelajari pola dari data sekuensial yang digunakan.

Selama proses pelatihan, model LSTM menunjukkan perkembangan akurasi dan loss yang stabil. Perubahan performa model dari epoch awal hingga epoch ke-100 dapat diamati melalui kurva akurasi dan loss yang ditampilkan pada Gambar 7 dan 8.



Gambar 7. Kurva training dan validation accuracy model LSTM

Pada Gambar 7 terlihat bahwa nilai training accuracy mengalami peningkatan konsisten sejak epoch awal hingga mendekati konvergensi pada sekitar epoch ke-70. Validation accuracy juga menunjukkan pola kenaikan yang relative sejalan dengan training accuracy. Pada epoch ke-100 training accuracy mencapai 92,91%, sementara validation accuracy sebesar 91,01%. Perbedaan yang relatif kecil ini menunjukkan bahwa model meiliki kemampuan generalisasi yang cukup baik.



Gambar 8. Kurva training dan validation loss model LSTM

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

Gambar 8 memperlihatkan penurunan nilai loss pada kedua kurva yang berlangsung secara konsisten hingga akhir pelatihan. Training loss menurun pada epoch awal dan berjalan dengan stabil. Sedangkan validation loss juga menunjukkan pola penurunan yang serupa dengan tingkat fluktuasi yang rendah. Pada epoch ke-100, nilai training loss berada pada 0,23 dan validation loss pada 0,25. Indikasi model tidak mengalami overfitting yang signifikan diperkuat dengan kestabilan performa antara data train dan validation serta selisih yang relatife kecil antara kedua loss curve.

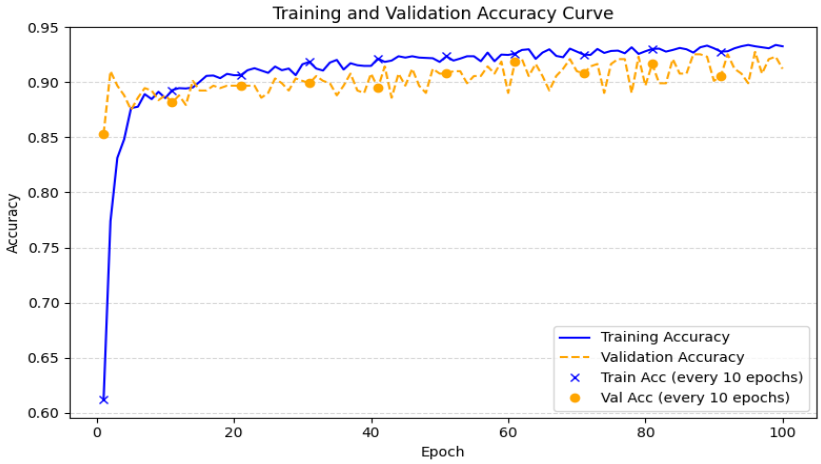
3.3.2 Bi-LSTM

Model BiLSTM yang digunakan pada penelitian ini dirancang untuk memproses data sekuensial dari dua arah, yaitu forward dan backward, sehingga mampu memanfaatkan konteks temporal baik dari masa lalu maupun masa depan dalam satu rangkaian sekuens. Arsitektur model terdiri dari dua lapisan Bidirectional LSTM berurutan, Batch Normalization, lapisan Dense dengan aktivasi ReLU, serta dua lapisan Dropout untuk regularisasi. Selain itu, L2 Regularization diterapkan pada lapisan utama untuk mencegah bobot berlebih, dan lapisan output Sigmoid digunakan untuk menghasilkan probabilitas klasifikasi biner. Arsitektur lengkap ditunjukkan pada Tabel 3.

Tabel 3. Arsitektur Model BiLSTM

Layer	Input	Output	Parameter
Input	(None, 90, 2)	(None, 90, 2)	
Bidirectional LSTM	(None, 90, 2)	(None, 90, 128)	34.304
Batch Normalization	(None, 90, 128)	(None, 90, 128)	512
Bidirectional LSTM	(None, 90, 128)	(None, 64)	41.216
Dropout	(None, 64)	(None, 64)	0
Dense	(None, 64)	(None, 32)	2.080
Dropout	(None, 32)	(None, 32)	0
Dense	(None, 32)	(None, 1)	33

Detail konfigurasi pada Tabel 3 menunjukkan bahwa lapisan Bidirectional LSTM pertama memiliki 34.304 parameter yang digunakan untuk mengekstraksi pola sekuensial dari arah maju dan mundur secara parallel. Batch Normalization dengan 256 parameter membantu menstabilkan distribusi input. Lapisan Bidirectional LSTM kedua dengan jumlah parameter sebanyak 41.216 menghasilkan representasi yang lebih padat, sementara lapisan Dense dengan 2.080 parameter dan activation function ReLU berperan dalam memproyeksikan hasil ke dimensi klasifikasi. Pada lapisan Bidirectional maupun Dense, diterapkan L2 Regularization untuk memberikan penalty pada bobot yang berlebih. Selain itu, dua lapisan Dropout dengan dropout rate sebesar 0,5 turut digunakan utnuk mengurangi fluktuasi dan overfitting saat proses training model. Lapisan output Sigmoid dengan 1 unit neuron menghasilkan porbabilitas prediksi biner antara normal atau yawning. Jumlah trainable parameter yang digunakan dalam proses BiLSTM yaitu 77.889. Hal ini menunjukkan bahwa model BiLSTM lebih kompleks dibandingkan LSTM, namun kompleksitas tambahan tersebut memberikan kemampuan representasi temporal yang lebih kaya jika dibandingkan LSTM konvensional. Selama proses pelatihan, model Bidirectional LSTM menunjukkan perkembangan akurasi dan loss yang lebih stabil jika dibandingkan dengan LSTM konvensional. Perubahan performa model dari epoch awal hingga epoch ke-100 dapat diamati melalui kurva akurasi dan loss yang ditampilkan pada Gambar 9 dan 10.

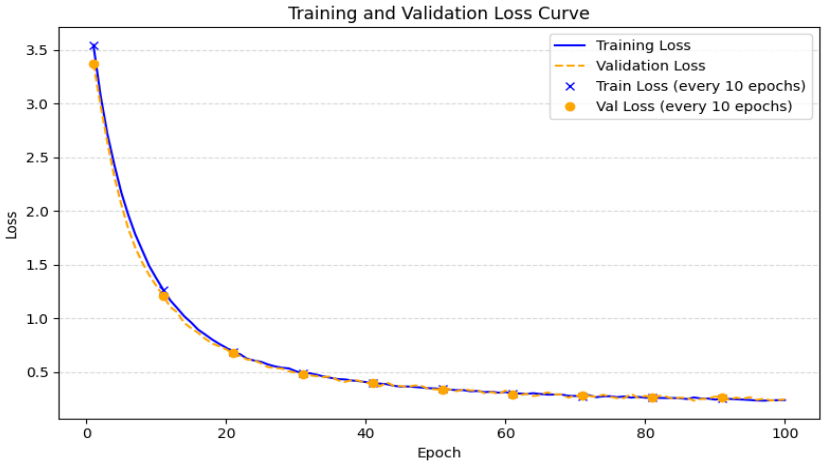


Gambar 9. Kurva training dan validation accuracy model BiLSTM

Pada Gambar 9 terlihat bahwa akurasi model BiLSTM meningkat stabil sejak awal proses training dan mulai mencapai titik konvergensi pada sekitar epoch ke-60. Pola peningkatan pada kurva training dan validation relative konsisten dengan fluktuasi yang minim pada validation accuracy. Hingga epoch ke-100, training accuracy tercatat sebesar 93,27%, sementara validation accuracy mencapai 91,23%. Selisih yang kecil antara keduanya menandakan

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

bahwa model tidak mengalami overfitting yang signifikan serta memiliki kemampuan generalisasi yang baik. Hal ini membuktikan bahwa pemrosesan dua arah pada BiLSTM membantu memperkuat representasi temporal sehingga akurasi pada validation dapat terjaga dengan stabil.

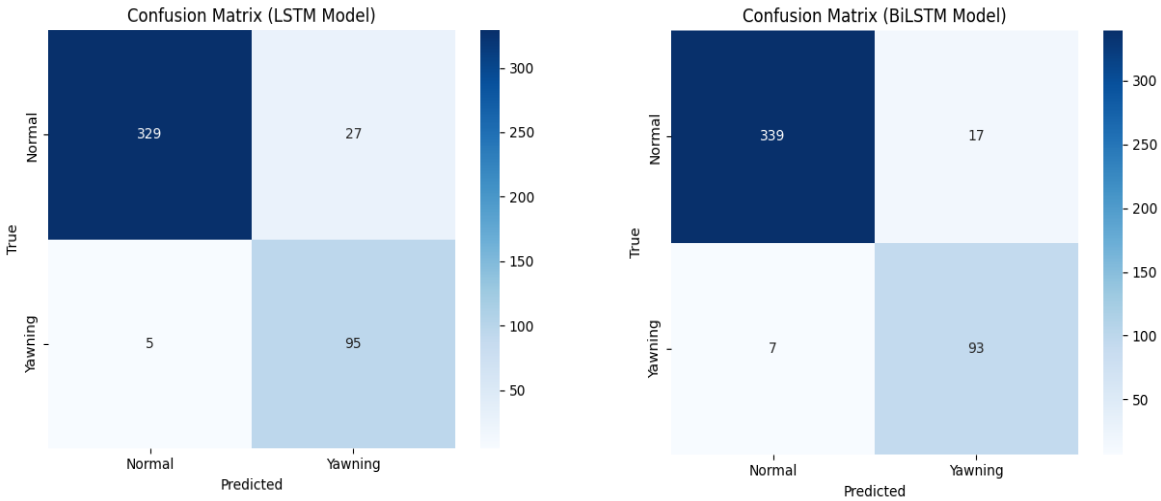


Gambar 10. Kurva training dan validation loss model BiLSTM

Gambar 10 memperlihatkan bahwa nilai loss pada kurva training dan validation mengalami penurunan yang konsisten hingga mendekati konvergensi di epoch akhir. Training loss pada epoch ke-100 tercatat sebesar 0,24, sementara validation loss juga sebesar 0,24. Nilai yang relatif serupa memperkuat indikasi bahwa model BiLSTM mampu menjaga keseimbangan performa antara data train dan data validation. Kurva loss pada BiLSTM terlihat lebih stabil dibandingkan dengan LSTM. Hal ini menunjukkan bahwa kemampuan pemrosesan informasi secara dua arah dapat membantu model dalam mengenali pola urutan waktu secara lebih konsisten. Sehingga, fluktuasi selama pelatihan dapat diminimalkan.

3.4 Hasil Evaluasi Model

Pada tahapan ini evaluasi model dilakukan terhadap data uji, yang diawali dengan mengidentifikasi matrik dari *confussion matrix* untuk memperoleh nilai *accuracy*, *precision*, *recall*, dan *F1-score* pada *weighted average*. *Confusion matrix* memberikan gambaran rinci mengenai distribusi klasifikasi benar maupun salah pada masing-masing kelas. Hasil *confusion matrix* pada kedua model dapat dilihat pada Gambar 11.



(a) (b)

Gambar 11. Confusion Matrix model (a) LSTM dan (b) BiLSTM

Pada model LSTM, dari 356 data kelas normal, 329 diprediksi benar dan 27 salah ke yawning; sedangkan dari kelas yawning, 95 benar dan 5 salah ke normal. Pada model BiLSTM, akurasi lebih seimbang: 339 data normal benar (17 salah), dan 93 data yawning benar (7 salah). Model BiLSTM menunjukkan kinerja yang lebih stabil dibanding LSTM karena mampu menjaga keseimbangan antara prediksi pada kelas normal dan yawning. Meskipun keduanya memiliki performa baik, BiLSTM lebih unggul dengan kesalahan klasifikasi yang lebih sedikit dan distribusi prediksi yang lebih proporsional. Kemudian untuk performa kedua model dapat dilihat pada Tabel 4 berikut.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

Tabel 4. Perbandingan performa model LSTM dan BiLSTM

Model	Accuracy	Precision	Recall	F1-score
LSTM	92,98%	93,98%	92,98%	93,22%
BiLSTM	94,74%	95,03%	94,74%	94,82%

Hasil pada Tabel 4 menunjukkan bahwa model LSTM mencapai akurasi 92,98% dengan precision 93,98%, recall 92,98%, dan F1-score 93,22%. Sementara itu, BiLSTM memberikan hasil lebih baik dengan akurasi 94,74%, precision 95,03%, recall 94,74%, dan F1-score 94,82%. Peningkatan ini menunjukkan bahwa BiLSTM lebih konsisten dalam mengenali kelas mayoritas (normal) maupun minoritas (yawning), dengan tingkat false positives dan false negatives yang lebih rendah. Keunggulan BiLSTM berasal dari kemampuannya memanfaatkan konteks temporal dua arah, sehingga lebih efektif dalam mengenali pola dinamis seperti menguap atau menutup mata. Selain itu, BiLSTM juga lebih stabil dalam generalisasi, yang terlihat dari perbedaan metrik training dan validation yang kecil. Perbandingan dengan penelitian terdahulu menggunakan dataset serupa ditunjukkan pada Tabel 5.

Tabel 5. Perbandingan Hasil Penelitian

Penelitian	Algoritma	Akurasi
Long et al.	LSTM	88,00%
Jing et al.	2s-STGCN	93,40%
Our*	BiLSTM	94,74%

Tabel 5 memperlihatkan bahwa performa BiLSTM pada penelitian ini lebih tinggi dibandingkan dengan hasil kedua penelitian lainnya. Dengan capaian akurasi 94,74%, penelitian ini memberikan kontribusi praktis berupa peningkatan performa deteksi kantuk pengemudi berbasis data sekuensial.

Meskipun hasil yang diperoleh cukup baik, penelitian ini masih memiliki keterbatasan. Fitur yang digunakan hanya mencakup EAR dan MOR sehingga belum menggambarkan tanda kantuk lain seperti pergerakan kepala atau ekspresi wajah yang lebih kompleks. Selain itu, dataset yang digunakan masih terbatas pada kondisi dan kelas tertentu.

4. KESIMPULAN

Penelitian ini berhasil membuktikan bahwa penggunaan algoritma LSTM dan BiLSTM efektif untuk mendeteksi kantuk pengemudi berdasarkan fitur sekuensial wajah berupa EAR dan MOR. Kedua model menujukkan performa yang baik, namun hasil pengujian memperlihatkan bahwa BiLSTM lebih unggul dibandingkan LSTM. BiLSTM mampu mencapai akurasi 94,47% dan F1-score 94,82%, sementara LSTM hanya mencapai akurasi 92,98% dan F1-score 93,22%. Peningkatan ini menegaskan bahwa pemrosesan data dua arah pada BiLSTM memberikan kemampuan yang lebih kuat dalam menangkap pola temporal yang kompleks, sehingga model dapat mengidentifikasi gejala kantuk seperti menutupnya mata atau menguap dengan lebih akurat. Secara praktis, temuan ini berkontribusi pada pengembangan sistem deteksi kantuk berbasis visi computer yang lebih handal dan dapat diimplementasikan dalam teknologi keselamatan berkendara. Implikasi penelitian ini menunjukkan bahwa integrasi BiLSTM dalam sistem monitoring pengemudi dapat membantu mengurangi risiko kecelakaan akibat kantuk. Kedepannya, penelitian dapat dikembangkan dengan memperkaya fitur melalui pendekatan multimodal serta mengeksplorasi arsitektur deep learning lain untuk menghasilkan model yang lebih adaptif dan akurat dalam kondisi nyata.

REFERENCES

[1] B. P. S. Indonesia, "Jumlah Kecelakaan, Korban Mati, Luka Berat, Luka Ringan, dan Kerugian Materi - Tabel Statistik," https://www.bps.go.id/id/statistics-table/2/NTEzIzI=/jumlah-kecelakaan-korban-mati-luka-berat-luka-ringan-dan-kerugian-materi.html.

[2] R. Aprianto, A. Rokhim, A. Basuki, and S. Sugiyarto, "Pengaruh Karakteristik Pengemudi Dan Pemanfaatan Rest Area Terhadap Kelelahan Pengemudi Studi Kasus Ruas Jalan Tol Pejagan - Solo," *Jurnal Keselamatan Transportasi Jalan (Indonesian Journal of Road Safety)*, vol. 8, pp. 92–103, May 2021, doi: 10.46447/ktj.v8i1.310.

[3] V. A. Saeed, "A Framework for Recognition of Facial Expression Using HOG Features," *International Journal of Mathematics, Statistics, and Computer Science*, vol. 2, pp. 1–8, May 2024, doi: 10.59543/IJMSCS.V2I.7815.

[4] T. Tinaliah and T. Elizabeth, "Penerapan Convolutional Neural Network Untuk Klasifikasi Citra Ekspresi Wajah Manusia Pada MMA Facial Expression Dataset," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 8, no. 4, pp. 2051–2059, Dec. 2021, doi: 10.35957/JATISI.V8I4.1437.

[5] L. Chen, G. Xin, Y. Liu, and J. Huang, "Driver Fatigue Detection Based on Facial Key Points and LSTM," *Security and Communication Networks*, vol. 2021, pp. 1–9, Jun. 2021, doi: 10.1155/2021/5383573.

[6] M. F. Rizkilloh and S. Widiyanesti, "Prediksi Harga Cryptocurrency Menggunakan Algoritma Long Short Term Memory (LSTM)," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 6, pp. 25–31, Feb. 2022, doi: 10.29207/resti.v6i1.3630.

[7] Z. Ruiye, "Volleyball training video classification description using the BiLSTM fusion attention mechanism," *Heliyon*, vol. 10, no. 15, p. e34735, Aug. 2024, doi: 10.1016/J.HELIYON.2024.E34735.

[8] Y. Albadawi, A. AlRedhaei, and M. Takruri, "Real-Time Machine Learning-Based Driver Drowsiness Detection Using Visual Features," *Journal of Imaging 2023, Vol. 9, Page 91*, vol. 9, no. 5, p. 91, Apr. 2023, doi: 10.3390/JIMAGING9050091.

Building of Informatics, Technology and Science (BITS)
Volume 7, No 2, September 2025 Page: 1343−1354
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v7i2.8219

[9] S. Abtahi, M. Omidyeganeh, S. Shirmohammadi, and B. Hariri, "YawDD: Yawning Detection Dataset," *IEEE Dataport*, pp. 24–28, Aug. 2020, doi: 10.21227/e1qm-hb90.
[10] J. R. González-Rodríguez, D. M. Córdova-Esparza, J. Terven, and J. A. Romero-González, "Towards a Bidirectional Mexican Sign Language–Spanish Translation System: A Deep Learning Approach," *Technologies (Basel)*, vol. 12, no. 1, Jan. 2024, doi: 10.3390/TECHNOLOGIES12010007.
[11] D. Kantuk Untuk Keamanan Berkendara Berbasis Pengolahan Citra, C. Kurniawan Umbu Nggiku, and A. Rabi, "Deteksi Kantuk Untuk Keamanan Berkendara Berbasis Pengolahan Citra," *Jurnal JEETech*, vol. 4, no. 1, pp. 48–56, Sep. 2023, doi: 10.32492/JEETECH.V4I1.4107.
[12] T. Zhu *et al.*, "Research on a Real-Time Driver Fatigue Detection Algorithm Based on Facial Video Sequences," *Applied Sciences 2022, Vol. 12, Page 2224*, vol. 12, no. 4, p. 2224, Feb. 2022, doi: 10.3390/APP12042224.
[13] G. Huang, D. Wang, Y. Du, Q. Zhang, Z. Bai, and C. Wang, "Deformation Feature Extraction for GNSS Landslide Monitoring Series Based on Robust Adaptive Sliding-Window Algorithm," *Front Earth Sci (Lausanne)*, vol. 10, Apr. 2022, doi: 10.3389/feart.2022.884500.
[14] F. Hamidy and I. Yasin, "Penerapan Metode Moving Average Dalam Penentuan Harga Pokok Penjualan Barang Berbasis Web," *CHAIN: Journal of Computer Technology, Computer Engineering, and Informatics*, vol. 2, pp. 67–76, Apr. 2024, doi: 10.58602/chain.v2i2.115.
[15] E. Sutanty, Maukar, D. K. Astuti, and Handayani, "Penerapan Model Arsitektur VGG16 Untuk Klasifikasi Jenis Sampah," *Decode: Jurnal Pendidikan Teknologi Informasi*, vol. 3, no. 2, pp. 407–419, Sep. 2023, doi: 10.51454/DECODE.V3I2.331.
[16] B. Deepa and K. Ramesh, "Epileptic seizure detection using deep learning through min max scaler normalization," *Int J Health Sci (Qassim)*, pp. 10981–10996, May 2022, doi: 10.53730/ijhs.v6ns1.7801.
[17] Muljono, S. A. Wulandari, H. Al Azies, M. Naufal, W. A. Prasetyanto, and F. A. Zahra, "Breaking Boundaries in Diagnosis: Non-Invasive Anemia Detection Empowered by AI," *IEEE Access*, vol. 12, pp. 9292–9307, 2024, doi: 10.1109/ACCESS.2024.3353788.
[18] Karthiek. Reddy Bokka, Shubhangi. Hora, Tanuj. Jain, and Monicah. Wambugu, *Deep Learning for Natural Language Processing : Solve Your Natural Language Processing Problems with Smart Deep Neural Networks*. Birmingham: Packt Publishing, Limited, 2019.
[19] G. Fajar Shidik *et al.*, "LUTanh Activation Function to Optimize BI-LSTM in Earthquake Forecasting," *International Journal of Intelligent Engineering and Systems*, vol. 17, no. 1, 2024, doi: 10.22266/ijies2024.0229.48.
[20] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive Into Deep Learning*. Cambridge University Press, 2023.
[21] Z. Hameed and B. Garcia-Zapirain, "Sentiment Classification Using a Single-Layered BiLSTM Model," *IEEE Access*, vol. 8, pp. 73992–74001, Apr. 2020, doi: 10.1109/ACCESS.2020.2988550.
[22] M. Naufal, H. Al Azies, and R. M. Brilianto, "Enhanced Brain Tumor Classification through Gamma Correction in Deep Learning," *SISTEMASI*, vol. 13, no. 6, pp. 2348–2358, Nov. 2024, doi: 10.32520/STMSI.V13I6.4474.
[23] I. K. Trisiawan, Y. Yuliza, F. Supegina, and S. Attamimi, "Penerapan Multi-Label Image Classification Menggunakan Metode Convolutional Neural Network (CNN) Untuk Sortir Botol Minuman," *Jurnal Teknologi Elektro*, vol. 13, no. 1, pp. 48–54, Feb. 2022, doi: 10.22441/JTE.2022.V13I1.009.
[24] S. Clara *et al.*, "Implementasi Seleksi Fitur pada Algoritma Klasifikasi Machine Learning untuk Prediksi Penghasilan pada Adult Income Dataset," *Prosiding Seminar Nasional Mahasiswa Bidang Ilmu Komputer dan Aplikasinya*, vol. 2, no. 1, pp. 741–747, Jul. 2021, Accessed: Aug. 06, 2025. [Online]. Available: https://conference.upnvj.ac.id/index.php/senamika/article/view/1417
[25] S. Abtahi, M. Omidyeganeh, S. Shirmohammadi, and B. Hariri, "YawDD: A yawning detection dataset," *Proceedings of the 5th ACM Multimedia Systems Conference, MMSys 2014*, pp. 24–28, 2014, doi: 10.1145/2557642.2563678.
[26] C. Aj. Saputra, D. Erwanto, and P. N. Rahayu, "Deteksi Kantuk Pengendara Roda Empat Menggunakan Haar Cascade Classifier Dan Convolutional Neural Network," *JEECOM Journal of Electrical Engineering and Computer*, vol. 3, no. 1, pp. 1–7, Apr. 2021, doi: 10.33650/JEECOM.V3I1.1510.
[27] A. Julianto, A. Sunyoto, D. Ferry, and W. Wibowo, "Optimasi Hyperparameter Convolutional Neural Network Untuk Klasifikasi Penyakit Tanaman Padi," *TEKNIMEDIA: Teknologi Informasi dan Multimedia*, vol. 3, no. 2, pp. 98–105, Dec. 2022, doi: 10.46764/TEKNIMEDIA.V3I2.77.