BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Beberapa penelitian menyoroti pentingnya pengembangan teknologi yang dapat menerjemahkan bahasa isyarat ke dalam teks atau suara guna meningkatkan inklusivitas komunikasi. Sebuah penelitian mengembangkan kerangka kerja untuk pengenalan bahasa isyarat dengan mengombinasikan CNN dan *Bidirectional long short-term memory* (Bi-LSTM) guna mempelajari dinamika *temporal* dalam urutan video. Dataset yang digunakan mencakup RWTH-PHOENIX-Weather 2014 dan SIGNUM, dengan metode *DeepFlow* untuk menghitung *optical flow*. Model ini dilatih sebanyak 100 *epoch* dan menghasilkan *word error rate* (WER) sebesar 22,86%, menunjukkan peningkatan kinerja yang signifikan dibandingkan modalitas tunggal serta mengungguli metode *state-of-the-art* dengan peningkatan lebih dari 15% (Cui dkk., 2019).

Penelitian lainnya membandingkan dua model CNN yang dibuat khusus untuk mengenali huruf-huruf dalam *American Sign language* (ASL). Penelitian ini bertujuan untuk menentukan model mana yang memiliki kinerja lebih baik dalam mengenali 26 tanda tangan yang berbeda. Kedua model menggunakan kombinasi CNN dan fungsi aktivasi *softmax,* tetapi dengan *hyperparameter* yang berbeda pada *hidden layer*nya, yaitu pada Model_1 digunakan *dropout-rate* 0,5 dan fungsi aktivasi *Relu*, sedangkan Model_2 digunakan fungsi aktivasi *Relu* saja. Dataset yang digunakan terdiri dari 78,000 gambar huruf ASL dalam format RGB, dengan setiap huruf memiliki 3000 gambar. Hasil penelitian menunjukkan bahwa Model_2 memiliki kinerja keseluruhan yang lebih baik dibandingkan Model_1, dengan akurasi 98,44% dan skor F1 98,41%. Namun, kinerja setiap model bervariasi tergantung pada huruf tertentu, menunjukkan bahwa pilihan model mungkin bergantung pada kasus penggunaan spesifik dan huruf yang diminati. Penelitian ini menyoroti pentingnya perancangan model yang disesuaikan dengan dataset tertentu serta menunjukkan bahwa kombinasi CNN dan fungsi *softmax*

memiliki potensi dalam meningkatkan kinerja pengenalan bahasa isyarat (Rakshit dkk., 2024).

Penelitian terkait juga membangun pendekatan pengenalan bahasa isyarat berbasis data sensor dan *deep learning* untuk meningkatkan akurasi pengenalan bahasa isyarat. Penelitian ini merancang jaringan CNN satu dimensi menggunakan pendekatan *skip connection* dan menggabungkan mekanisme *multi-head attention* yang ditingkatkan dengan jaringan *Bidirectional long short-term memory* (BiLSTM). Arsitektur yang dihasilkan, disebut BMCNN, diuji melalui validasi silang sepuluh kali lipat pada dataset sensor dan mencapai akurasi 99,13%. Hasil penelitian menunjukkan bahwa pendekatan ini mengungguli teknik pembelajaran mesin tradisional dan metode mutakhir lainnya dalam bidang ini. Penelitian ini menyoroti pentingnya kombinasi data sensor dan *deep learning* untuk meningkatkan akurasi pengenalan bahasa isyarat, serta potensi penggunaan mekanisme *multi-head attention* dan BiLSTM dalam menangkap fitur *temporal* dan spasial dari gerakan isyarat (Hao dkk., 2024).

Penelitian yang bertujuan untuk membangun sistem pengenalan bahasa isyarat Arab statis yang terdiri dari 220,000 gambar RGB yang mewakili 44 kategori, termasuk 32 huruf Arab serta angka dari 0 hingga 10, menggunakan model CNN juga telah dilakukan. Fokus utama penelitian adalah meningkatkan akurasi pengenalan bahasa isyarat melalui pendekatan multi-model untuk memperkuat proses ekstraksi fitur dan klasifikasi. Digunakan berbagai model *pre-trained* seperti *Dense*Net121, VGG16, ResNet50, MobileNetV2, Xception, EfficientNet B0, NASNet *Mobile*, dan InceptionV3. Hasil terbaik diperoleh dari kombinasi *Dense*Net121 dan VGG16, yang mencapai akurasi 100% pada dataset pengujian. Dibandingkan dengan model tunggal, multi-model menunjukkan performa yang lebih baik dalam hal akurasi, presisi, *recall*, dan *F1-score*. Kesimpulan dari penelitian ini adalah bahwa metode multi-model efektif dalam meningkatkan performa pengenalan bahasa isyarat Arab statis. (Ismail dkk., 2021)

Penelitian lainnya juga melakukan eksperimen komparatif pada beberapa metode berbasis visi komputer untuk pengenalan bahasa isyarat, dengan menggunakan dataset publik dan model *deep learning*. Fokus penelitian ini

memetakan aliran video yang tidak tersegmentasi ke dalam *glosses* atau unit kata dalam bahasa isyarat, serta mengembangkan dataset RGB+D baru untuk *Greek Sign language* (GSL). Dataset ini mencakup tiga tingkat anotasi: *gloss* individu, kalimat, dan terjemahan ke bahasa lisan. Penelitian ini menunjukkan bahwa 3D-CNN lebih efektif untuk pengenalan bahasa isyarat terisolasi, juga memberikan hasil yang lebih baik untuk pengenalan bahasa isyarat berkelanjutan (Adaloglou dkk., 2021).

Dapat dikatakan, metode 3D-CNN dapat bekerja dengan baik dalam hal ekstraksi fitur dari data video yang berkelanjutan. Seiring perkembangannya, dikembangkan teknologi baru berdasarkan 3D-CNN, contohnya seperti *MediaPipe*, penelitian oleh (Sánchez-Vicinaiz dkk., 2024) mengembangkan model untuk mendeteksi *fingerspelling* dalam *Mexican Sign language* (MSL) menggunakan *MediaPipe* dan CNN. Model ini dirancang untuk mengenali tanda-tanda statis dari alfabet *dactyl logical* MSL dengan menggunakan *MediaPipe* untuk ekstraksi fitur dan model CNN untuk klasifikasi. Dataset yang digunakan terdiri dari 3822 gambar tangan yang diambil dari satu individu, dengan setiap gambar memenuhi kriteria tertentu seperti pencahayaan yang baik dan posisi tangan yang jelas. Model CNN terbaik mencapai akurasi 83,63% pada set pengujian yang terdiri dari 336 gambar. Penelitian ini menunjukkan bahwa penggunaan *MediaPipe* untuk menandai titik-titik struktural pada tangan dapat meningkatkan akurasi deteksi dan memungkinkan implementasi model pada perangkat dengan konsumsi daya rendah, seperti *Raspberry Pi*, untuk klasifikasi langsung. Hasil ini menunjukkan potensi besar dalam meningkatkan aksesibilitas dan pembelajaran bahasa isyarat melalui teknologi pengenalan gambar.

Sehingga terdapat penelitian yang mengembangkan model pengenalan *Assamese Sign language* secara langsung menggunakan *MediaPipe* dan *deep learning*. Penelitian ini bertujuan untuk mengenali beberapa huruf dasar dari alfabet *Assamese Sign language*. Dataset yang digunakan terdiri dari 2094 data poin yang mencakup 9 gerakan statis dengan vokal dan konsonan dari *Assamese Sign language*. *MediaPipe* digunakan untuk mendeteksi titik-titik *landmark* pada gambar tangan, yang kemudian dinormalisasi dan disimpan. Model jaringan saraf

tiruan *feedforward* dilatih menggunakan data ini dan mencapai akurasi 99%. Hasil penelitian menunjukkan bahwa metode ini efektif untuk mengenali huruf dan gerakan lain dalam bahasa isyarat *Assamese Sign language*. Metode ini juga dapat diterapkan untuk pengenalan tanda dan gerakan dalam berbagai bahasa isyarat lokal lainnya di India (Bora dkk., 2022)

MediaPipe juga dimanfaatkan ke dalam model LSTM, seperti penelitian yang mengembangkan model *Continuous Sign language recognition* (CSLR) menggunakan *MediaPipe holistic* dan LSTM, pada dataset *Indian Sign language* (ISL). Dataset yang terdiri dari 45 tanda dan gerakan ISL, termasuk alfabet dan frasa bahasa Inggris, dikumpulkan menggunakan webcam melalui *OpenCV* dan *Python*. Dengan menggunakan *MediaPipe holistic*, peneliti melacak gerakan wajah, tangan, dan tubuh untuk mengekstraksi *landmark* utama dari video secara langsung. Model LSTM yang dirancang memiliki enam lapisan, tiga lapisan LSTM dan tiga lapisan *Dense*, yang dilatih pada 95% data yang dikumpulkan. Hasil eksperimen menunjukkan akurasi pengenalan sebesar 88,23%, baik dalam pengujian data maupun pengujian langsung. Keberhasilan model ini ditingkatkan oleh penggunaan *MediaPipe*, yang mengurangi kompleksitas model dan mempercepat pelatihan, meskipun dataset yang digunakan relatif kecil (Srivastava dkk., 2024).

Penelitian serupa mengembangkan metode pengenalan *Indian Sign language* menggunakan *pipeline MediaPipe holistic* untuk ekstraksi fitur dan jaringan LSTM untuk klasifikasi. Metode ini dievaluasi pada dataset INCLUDE, yang berisi video ISL dengan skala besar, dan subsetnya yang lebih kecil yaitu INCLUDE-50. Hasilnya menunjukkan akurasi 94,8% pada INCLUDE-50 dan 87,4% pada INCLUDE, dengan skor F1 rata-rata makro masing-masing sebesar 93,5% dan 86,6%. Metode ini mengungguli kinerja *state-of-the-art* pada dataset tersebut. Penelitian ini juga menyoroti pentingnya augmentasi data untuk meningkatkan generalisasi model, terutama dalam menangani kelas dengan kemiripan tinggi antar kelas. Dengan menggunakan *MediaPipe holistic*, penelitian ini berhasil mengekstraksi 258 titik kunci dari setiap *frame* video, yang kemudian

digunakan untuk pelatihan model LSTM, menghasilkan model pengenalan bahasa isyarat yang efisien dan akurat (Khartheesvar dkk., 2024).

Penelitian serupa lainnya juga mengembangkan model pengenalan ASL secara langsung menggunakan *deep learning* LSTM dan deteksi tangan dengan *MediaPipe*. Model ini dirancang untuk mengenali gerakan tangan secara langsung dari video tanpa memerlukan perangkat eksternal. Dataset yang digunakan mencakup 100 tanda yang dilakukan oleh satu penanda dalam tiga kondisi pencahayaan dan kecepatan yang berbeda. Setiap video dipecah menjadi *frame* dan ditingkatkan menjadi 2400 gambar pertanda. Model LSTM digunakan untuk memprediksi tanda yang ditampilkan dari serangkaian *frame* video. *MediaPipe* digunakan untuk mendeteksi dan melacak *landmark* tangan, menghasilkan 21 *landmark* tangan yang digunakan sebagai *input* model LSTM. Model mencapai akurasi 99,35% setelah pelatihan dan implementasi. Penelitian ini juga menyoroti pentingnya penggunaan *dropout* untuk mengatasi *overfitting* dalam model *deep learning*, serta penggunaan latar belakang minimalis dan pencahayaan yang konsisten untuk meningkatkan akurasi deteksi (Abdulhamied dkk., 2023). Daftar publikasi yang terkait dengan penelitian ini dapat dilihat pada Tabel 2.1.

Tabel 2. 1. Daftar penelitian terkait

No	Penulis	Dataset	Algoritma	Keterangan
1	Cui dkk., 2019	*American sign language*	CNN dan Bi-LSTM	Menggunakan CNN dan Bi-LSTM pada dataset RWTH-PHOENIX-Weather 2014 dan SIGNUM. Menghasilkan WER sebesar 22,86%, unggul lebih dari 15% dibanding metode sebelumnya.
2	Rakshit dkk., 2024	*American sign language*	CNN dan fungsi aktivasi s*oftmax*	Membandingkan dua model CNN untuk mengenali huruf ASL dengan akurasi 98,44% dan *F1-score* 98,41%. Terdiri dari 78,000 gambar RGB huruf ASL. Model_2 lebih baik dalam mengenali tanda-tanda ASL.

Tabel 2. 1. Daftar penelitian terkait lanjutan

No	Penulis	Dataset	Algoritma	Keterangan
3	Hao dkk., 2024	*Chinese sign language*	BMCNN (CNN, BiLSTM & *Multi-head Attention*)	Mengembangkan model BMCNN yang merupakan gabungan dari CNN + BiLSTM + *Multi-head Attention*, mencapai akurasi 99,13%. Hasil menunjukkan kombinasi data sensor dan *deep learning* efektif untuk pengenalan bahasa isyarat.
4	Ismail dkk., 2021	*Arabic sign language*	Multi-Model CNN	Model multi-CNN dengan kombinasi *Dense*Net121 dan VGG16 memberikan akurasi 100% pada dataset pengujian. Multi-model lebih efektif dalam meningkatkan akurasi pengenalan bahasa isyarat Arab statis.
5	Adalogl ou dkk., 2021	*Greek sign language*	3D CNN dan 2D CNN	Melakukan eksperimen dengan dataset *Greek Sign language*, dihasilkan 3D-CNN efektif untuk pengenalan bahasa isyarat terisolasi, juga lebih baik untuk pengenalan bahasa isyarat berkelanjutan.
6	Sánchez Vicinaiz dkk., 2024	*Mexican sign language*	*MediaPipe* dan CNN	Menggunakan *MediaPipe* dan CNN untuk mendeteksi *fingerspelling* dalam *Mexican Sign language*. Model mencapai akurasi 83,63% dengan dataset berupa gambar sebanyak 3822.

Tabel 2. 1. Daftar penelitian terkait lanjutan

No	Penulis	Dataset	Algoritma	Keterangan
7	Bora dkk., 2022	*Assamese sign language*	*MediaPipe* dan *Feedforward Neural network*	Menerapkan *MediaPipe* dan *feedforward neural network* untuk pengenalan bahasa isyarat *Assamese* dengan akurasi 99% pada dataset. Model ini bekerja secara *real-time* dengan huruf dasar.
8	Srivastava dkk., 2024	*Indian sign language*	*MediaPipe holistic* dan LSTM	Model *Continuous Sign language recognition* (CSLR) untuk *Indian Sign language* menggunakan *MediaPipe* dan LSTM, dengan akurasi pengenalan 88,23%. *MediaPipe* mempercepat pelatihan dan mengurangi kompleksitas.
9	Khartheesvar dkk., 2024	*Indian sign language*	*MediaPipe holistic* dan LSTM	Menggunakan *MediaPipe holistic* untuk ekstraksi fitur dan LSTM untuk klasifikasi *Indian Sign language*. Model mencapai akurasi 94,8% pada INCLUDE-50 dan 87,4% pada dataset INCLUDE.
10	Abdulhamied dkk., 2023	*American sign language*	*MediaPie* dan LSTM	Model pengenalan *real-time* untuk ASL menggunakan *MediaPipe* dan LSTM, dengan menambahkan *dropout layer* pada *training* dan *testing*, mencapai akurasi 99,35%.

Berbagai penelitian yang telah dilakukan memiliki kesamaan dalam memanfaatkan teknologi *deep learning* untuk pengenalan bahasa isyarat. Model-model seperti CNN dan LSTM digunakan untuk menangkap dinamika spasial dan *temporal* dari gerakan isyarat. Selain itu, penelitian-penelitian ini juga menunjukkan peningkatan kinerja dengan mengadopsi 3D-CNN atau *pipeline* seperti *MediaPipe* untuk ekstraksi fitur dan model CNN atau LSTM untuk klasifikasinya. Dengan akurasi yang mencapai lebih dari 90%, studi-studi tersebut berhasil memperbaiki kinerja model pengenalan bahasa isyarat.

Sebagai bagian dari upaya mengembangkan teknologi pengenalan BISINDO, penting untuk melakukan tinjauan terhadap dataset yang sudah tersedia. Dataset-dataset ini perlu dievaluasi secara mendalam dari segi cakupan, kualitas, dan kelengkapannya. Dengan memahami kekuatan dan kelemahan masing-masing dataset tersebut, identifikasi celah yang perlu diisi melalui pengembangan dataset baru yang lebih representatif dan kontekstual dapat dilakukan. Pengamatan dan evaluasi terhadap dataset-dataset yang ada dijelaskan pada Tabel 2.2.

Tabel 2. 2. Evaluasi dataset yang ada

No	Nama Dataset	Sumber	Format Data	Deskripsi Konten	Kekurangan atau Gap
1	Sistem Isyarat Bahasa Indonesia (SIBI).	Kaggle	Gambar	Berisi gambar tangan untuk bahasa isyarat SIBI (Sistem Isyarat Bahasa Indonesia), Mencakup 26 kelas yang mewakili huruf alfabet A-Z.	Tidak spesifik ke BISINDO, hanya mencakup alfabet. Tidak ada variasi ekspresi atau konteks kata tertentu dalam BISINDO. Tidak ada variasi tangan dengan hanya menggunakan satu tangan.

Tabel 2. 2. Evaluasi dataset yang ada lanjutan

No	Nama Dataset	Sumber	Format Data	Deskripsi Konten	Kekurangan atau Gap
2	Bahasa Isyarat Indonesia (BISINDO) *Alphabets.*	Kaggle	Gambar	Berisi 26 kelas alfabet BISINDO, dengan 4 gambar untuk setiap latar.	Terbatas pada alfabet tanpa mencakup gestur kata atau frasa dalam BISINDO. Juga tidak ada ekspresi atau wajah dari pengguna.
3	Calvin *Computer vision Project*	Roboflow	Gambar	Berisi gambar berlabel representasi bahasa isyarat dalam BISINDO dan SIBI dalam 34 kelas kata.	Tidak konsisten pada satu jenis bahasa isyarat. hanya berupa gambar, dengan banyak variasi dalam satu label yang merupakan representasi gerakan-gerakan dari *frame-frame* video. Juga tidak ada variasi latar.
4	Terbisa *Computer vision Project*	Roboflow	Gambar	Dataset yang berisi gambar berlabel untuk 19 BISINDO umum.	Masih berupa gambar, yang memiliki kemiripan pada label frasa berbeda karena gerakan yang mirip. Sehinga bisa terjadi duplikasi dalam pengenalan.

Tabel 2. 2. Evaluasi dataset yang ada lanjutan

No	Nama Dataset	Sumber	Format Data	Deskripsi Konten	Kekurangan atau Gap
5	BISINDO *Sign language Computer vision Project*	Roboflow	Gambar	Dataset mencakup 1.350 gambar dan video dengan 26 kelas yang merupakan representasi alfabet BISINDO.	Hanya berupa alfabet, tanpa varasi dalam gambar juga latar yang ada.
6	Deteksi Bahasa Isyarat Indonesia *Computer vision Project*	Roboflow	Gambar	Dataset ini mencakup 4.633 gambar dengan 17 kelas, diantara nya 10 kelas kata, dan 7 sisanya kelas alfabet.	Tidak konsisten pada satu jenis bahasa isyarat, juga tidak lengkap dalam pengumpulan data yang ada.
7	*Hand Sign* BISINDO (C,I,L,O,U,V)	Kaggle	Gambar	Berisi gambar representasi 26 kelas huruf BISINDO dengan varasi latar dan tata letak tangan yaitu *background* gelap dan terang.	Variasi dari data sangat baik, sayangnya masih terbatas hanya untuk huruf BISINDO.

Tabel 2. 2. Evaluasi dataset yang ada lanjutan

No	Nama Dataset	Sumber	Format Data	Deskripsi Konten	Kekurangan atau Gap
8	BISINDO - Video Dataset	Kaggle	Video	Berisi rangkuman atau potongan tiga video youtube yang mana representasi dari 26 kelas alfabet BISINDO. Dengan varasi dari 3 video maka terdapat variasi dari peraga BISINDO itu sendiri.	Terbatas pada alfabet tanpa mencakup gestur kata atau frasa dalam BISINDO.

Sebagian besar dataset BISINDO yang tersedia saat ini hanya mencakup alfabet atau kosa kata dasar, tanpa banyak variasi gaya maupun konteks penggunaan. Selain itu, representasi gerakan dan frasa yang sering digunakan dalam komunikasi sehari-hari masih sangat terbatas. Jumlah video yang tersedia juga masih minim, sehingga belum memadai untuk mempelajari transisi gerakan antara satu kata atau frasa ke kata atau frasa lainnya. Untuk mengatasi kekurangan ini, diperlukan pengembangan dataset baru dengan karakteristik yang lebih kaya. Format data yang ideal berupa video yang merekam transisi gerakan antara kata dan frasa dengan berbagai durasi. Kosa kata yang dikumpulkan harus mencakup ragam kata umum, frasa, dan ekspresi kompleks guna memperluas cakupan komunikasi.

Sejalan dengan tujuan penelitian ini, pengembangan dataset baru diharapkan dapat mendorong penelitian lebih lanjut di bidang pengenalan bahasa isyarat. Penelitian sebelumnya juga menunjukkan bahwa pendekatan *deep learning*, khususnya menggunakan LSTM yang mampu menangkap hubungan *temporal*

dalam urutan video, sangat cocok untuk diaplikasikan. Penggunaan dataset yang berkualitas dan metode pengenalan yang tepat akan berdampak signifikan terhadap peningkatan akurasi model. Hal ini dapat membantu menciptakan model pengenalan BISINDO yang lebih efisien dan inklusif. Oleh karena itu, pembangunan dataset yang kredibel serta pengembangan model berbasis kombinasi teknik *deep learning* dan *pipeline MediaPipe* menjadi fokus utama. Penelitian ini diharapkan dapat meningkatkan aksesibilitas dan komunikasi bagi komunitas penyandang disabilitas di Indonesia, sekaligus memberikan kontribusi signifikan dalam mendukung inklusi sosial.

2.2 Dasar Teori

2.2.1. Fungsi Aktivasi

Fungsi aktivasi merupakan komponen penting dalam metode *neural network* yang memperkenalkan *non-linearitas* ke dalam model, memungkinkan jaringan untuk belajar dan merepresentasikan pola yang kompleks dalam data. Dalam konteks *recurrent neural networks* (RNN), pemilihan fungsi aktivasi yang tepat sangat penting untuk kinerja dan stabilitas model (Dubey dkk., 2022). Salah satu fungsi aktivasi yang sering digunakan dalam RNN yaitu *tangen hiperbolik* (*tanh*). Fungsi *tanh* memampatkan nilai *input* ke dalam rentang [−1, 1], menjadikannya terpusat pada nol dan cocok untuk memodelkan urutan dengan nilai positif dan negatif. Secara matematis, fungsi *tanh* direpresentasikan dalam Persamaan 2.1 (Dubey dkk., 2022).

$$tanh\,(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{2.1}$$

Fungsi ini cocok untuk memodelkan urutan dengan nilai positif dan negatif. Dimana fungsi ini membantu dalam menjaga stabilitas nilai dalam *hidden state* dan mengurangi masalah *vanishing gradient*, yang biasa terjadi dalam RNN. Selain fungsi *tanh*, adanya yang dinamakan dengan fungsi *sigmoid*, yang mengubah nilai *input* menjadi rentang antara 0 dan 1, ini mirip dengan *tanh*, tetapi menghasilkan nilai dalam rentang yang berbeda, sehingga berguna untuk masalah di mana *output* perlu ditafsirkan sebagai probabilitas, fungsi *sigmoid* direpresentasikan dalam Persamaan 2.2 (Dubey dkk., 2022)

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (2.2)$$

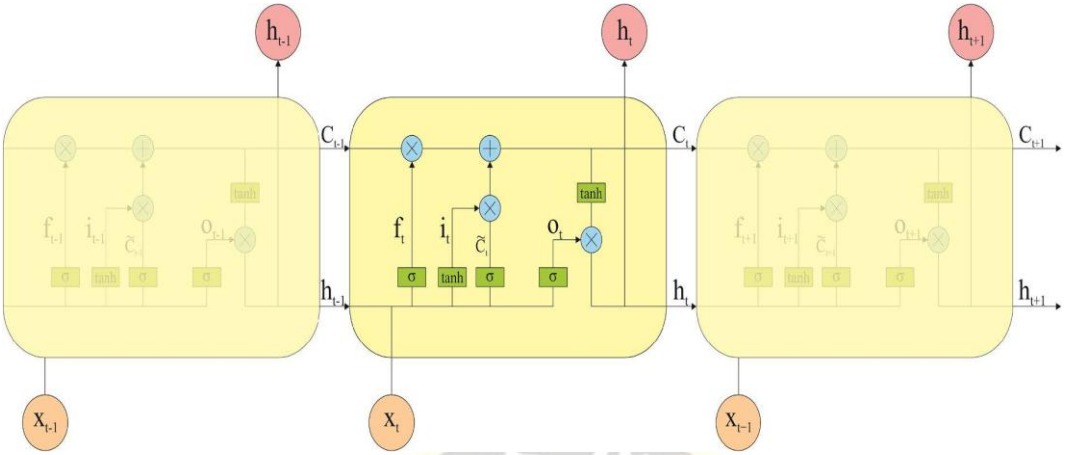
Fungsi aktivasi *sigmoid* sering digunakan di lapisan *output* untuk masalah klasifikasi *biner*. Ketika jaringan saraf digunakan untuk memprediksi apakah suatu *input* termasuk dalam salah satu dari dua kategori, yang mana berada dalam rentang 0 hingga 1, yang berarti fungsi ini sangat cocok untuk menghasilkan probabilitas atau klasifikasi *biner*, karena keluaran dapat ditafsirkan sebagai peluang. Demikian pula, fungsi *softmax* sering digunakan di lapisan *output* jaringan klasifikasi untuk mengkonversi skor mentah menjadi probabilitas, Fungsi ini sangat berguna dalam masalah klasifikasi multi-kelas. Fungsi *softmax* direpresentasikan secara matematis dalam Persamaan 2.3 (Dubey dkk., 2022).

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j \ e^{z_j}} \quad (2.3)$$

2.2.2. *Long short-term memory*

Long short-term memory (LSTM) yang mana jenis dari *recurrent neural networks* (RNN), dirancang khusus untuk memproses data sekuensial. LSTM diperkenalkan oleh Hochreiter dan Schmidhuber pada tahun 1997 dan telah menjadi salah satu model yang paling efektif untuk menangani data sekuensial yang memerlukan pemodelan ketergantungan jangka panjang, LSTM dapat mengatasi masalah *vanishing gradient* yang sering terjadi pada RNN biasa, sehingga dapat menangkap dependensi jangka panjang dalam data (Oropesa dkk., 2024). LSTM memiliki kemampuan yang sangat baik dalam mengingat informasi untuk jangka waktu yang lama, sehingga cocok untuk tugas-tugas seperti prediksi deret waktu, pengenalan ucapan, dan pemrosesan bahasa alami.

Struktur utama dari LSTM terdiri dari sebuah sel memori yang dikontrol oleh tiga gerbang utama, yaitu *forget gate*, *input gate*, dan *output gate*. *Input gate* yang menentukan informasi baru mana yang akan disimpan dalam sel memori, *forget gate* lalu mengontrol informasi mana yang akan dihapus dari sel memori, dan *output gate* akan menentukan bagian dari sel memori yang akan digunakan untuk menghasilkan *output*. Struktur tiga gerbang utama ini disebut dengan *cell state* (Chen dkk., 2020), gambar 2.1 memberikan gambaran visual tentang bagaimana ketiga gerbang ini bekerja bersama-sama dalam memproses informasi.



Gambar 2. 1. Struktur Gerbang LSTM (Chen dkk., 2020)

Dalam rangkaian prosesnya, setiap gerbang ini memainkan peran penting dalam mengatur aliran informasi melalui *cell state*. Berikut penjelasan rinci tentang setiap proses dalam *cell state* pada LSTM sesuai dengan urutan prosesnya.

2.2.2.1. *Forget Gate*

Forget gate bertugas sebagai gerbang yang mengatur aliran informasi ke dalam *cell state*. Ini memungkinkan LSTM untuk memutuskan informasi mana yang relevan dan harus disimpan untuk digunakan di masa depan, ini ditandai dengan persamaan *Forget gate* pada LSTM memainkan peran penting dalam mengatur aliran informasi dalam *cell state*. Dengan menentukan informasi mana yang harus dilupakan, *forget gate* membantu model untuk menghapus informasi yang tidak relevan dan mempertahankan informasi yang penting untuk prediksi masa depan. *Forget gate* secara matematis ditandai dengan Persamaan 2.4.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \quad (2.4)$$

f_t merepresentasikan nilai *output* dari *forget gate* pada waktu (t), σ (Sigma) merepresentasikan fungsi aktivasi *sigmoid*, W_f merepresentasikan bobot *forget gate* pada, $[h_{t-1}]$ merepresentasikan *hidden state* (ingatan jangka pendek) sebelumnya, x_t merepresentasikan *input* pada waktu (t), dan b_f merepresentasikan bias nya dalam, Nilai f_t yang mendekati 1 berarti sebagian besar informasi akan dipertahankan, sedangkan nilai yang mendekati 0 berarti sebagian besar akan dilupakan.

2.2.2.2. *Input gate*

Input gate bertugas sebagai gerbang yang mengatur aliran informasi ke dalam *cell state*. Ini memungkinkan LSTM untuk memutuskan informasi mana yang relevan dan harus disimpan untuk digunakan di masa depan, *Input Gate* ditandai dalam Persamaan 2.5.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.5)$$

i_t merepresentasikan nilai *output* dari *input gate* pada waktu (t), σ (Sigma) merepresentasikan fungsi aktivasi *sigmoid*, W_i merepresentasikan bobot *input gate* pada, $[h_{t-1}]$ merepresentasikan *hidden state* (ingatan jangka pendek) sebelumnya, x_t merepresentasikan *input* pada waktu (t), dan b_i merepresentasikan bias nya dalam, Nilai i_t yang tinggi menunjukkan bahwa banyak informasi baru akan ditambahkan. *Input gate* menggunakan fungsi aktivasi *sigmoid* untuk menentukan seberapa banyak informasi baru yang akan disimpan dalam *cell state*, dimana fungsi *sigmoid* mengubah nilai *input* menjadi rentang antara 0 dan 1, di mana 0 berarti tidak ada informasi yang disimpan dan 1 berarti semua informasi disimpan.

2.2.2.3. *Update Cell State*

Proses *update cell state* pada LSTM memungkinkan model untuk menyimpan informasi yang relevan dan menghapus informasi yang tidak relevan. Dimana prosesnya, yang pertama menghapus informasi yang tidak relevan, ini ditandai dengan Persamaan 2.6.

$$C_t = f_t * C_{t-1} \quad (2.6)$$

C_t merepresentasikan cell state pada waktu (t), f_t merepresentasikan *output* dari *forget state* pada waktu (t), dan C_{t-1} merepresentasikan *cell state* pada waktu sebelumnya, proses setelahnya menambahkan informasi yang baru yang relevan, ditandai dengan Persamaan 2.7.

$$C_t = C_t + i_t * \tilde{C}_t \quad (2.7)$$

C_t merepresentasikan *cell state* pada waktu (t), i_t *output* dari *input gate* pada waktu (t), dan $\tilde{C}_t$ merepresentasikan *candidate value* state pada waktu (t). *Candidate value* pada LSTM memainkan peran penting dalam menentukan informasi baru yang akan ditambahkan ke *cell state*, artinya ia menghitung nilai calon untuk update *cell state.* Dengan menggunakan fungsi aktivasi *tanh*,

candidate value membantu menjaga stabilitas nilai dalam *cell state* dan memastikan bahwa informasi yang relevan disimpan untuk digunakan di masa depan, selanjutnya dikombinasikan informasi yang lama dan baru tersebut, yang mana di tandai dalam Persamaan 2.8.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (2.8)$$

C_t merepresentasikan nilai *cell state* pada waktu (t), f_t merepresentasikan *output* dari *forget gate* pada waktu (t), C_{t-1} merepresentasikan *cell state* dari waktu sebelumnya, i_t merepresentasikan *output* dari *input gate* pada waktu (t), dan $\tilde{C}_t$ (*tilde* C) merepresentasikan *candidate value* pada waktu (t). *Candidate value* dikombinasikan dengan *output* dari *input gate* dan mengontrol seberapa banyak *candidate value* yang akan ditambahkan ke *cell state*. *Cell state* baru berupa kombinasi dari *cell state* sebelumnya yang mana telah dimodifikasi oleh *forget gate* dan informasi baru yang dipilih oleh *input gate*.

2.2.2.4. *Output Gate*

Output gate pada LSTM memainkan peran dalam mengatur aliran informasi yang keluar dari *cell state* dan digunakan untuk prediksi, dengan menggunakan fungsi aktivasi *sigmoid* dan *tanh*, dimana fungsi aktivasi *sigmoid* untuk menentukan seberapa banyak informasi dari *cell state* yang akan dikeluarkan, yang mana ditandai dengan Persamaan 2.9.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2.9)$$

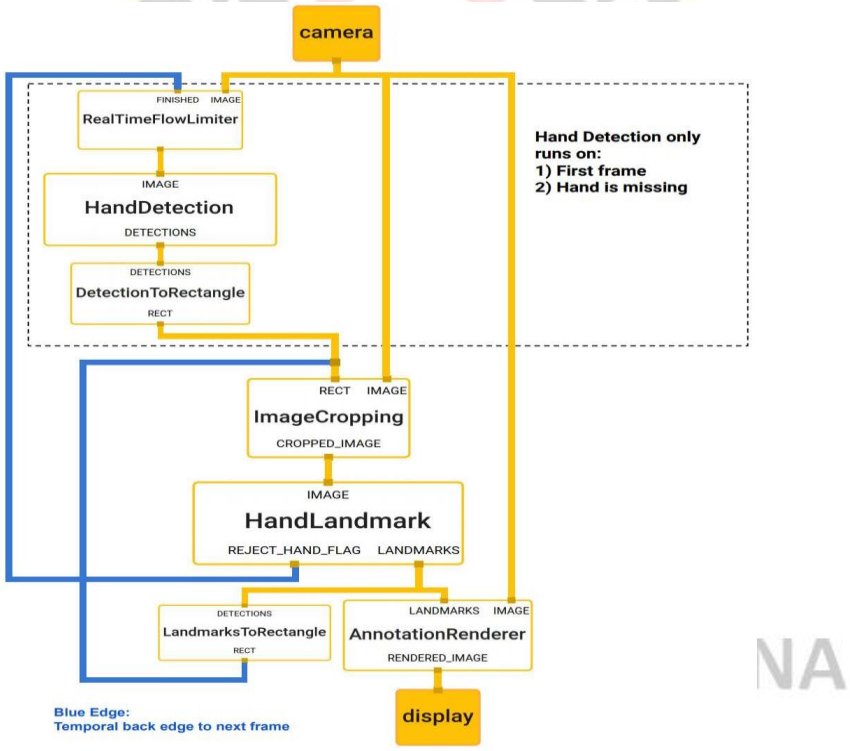
o_t merepresentasikan *output gate* pada waktu (t), σ (Sigma) merepresentasikan fungsi aktivasi *sigmoid*, W_o merepresentasikan beban dari *output gate*, h_{t-1} merepresentasikan *hidden state* sebelumnya, x_t merepresentasikan *input* pada waktu (t) dan b_o merepresentasikan bias dari *output gate*, dan proses setelahnya menambahkan informasi yang baru sebagai *hidden state*, ditandai dengan Persamaan 2.10.

$$h_t = o_t * tanh\,(C_t) \quad (2.10)$$

h_t merepresentasikan *hidden state* pada waktu (t), o_t merepresentasikan *output gate* pada waktu (t), dan $tanh\,(C_t)$ merepresentasikan *cell state* yang telah diperbarui dan diubah menggunakan fungsi aktivasi *tanh*.

2.2.3. *MediaPipe*

MediaPipe adalah *pipeline open source* yang dikembangkan oleh *Google* untuk menyediakan solusi *machine learning* yang dapat disesuaikan dengan berbagai tugas seperti deteksi objek, estimasi pose, deteksi wajah, pelacakan tangan, pelacakan iris dan lain sebagainya. Teknologi ini memungkinkan ekstraksi fitur dari video untuk memperoleh titik-titik *landmark* tubuh, seperti posisi pergelangan tangan dan ujung jari. *MediaPipe* juga memungkinkan model untuk mendeteksi tubuh meskipun dalam kondisi tertentu di mana sebagian tubuh terhalang atau tidak sepenuhnya terlihat (Sánchez-Vicinaiz dkk., 2024). Secara umum, proses pada *MediaPipe* terdiri dari beberapa tahapan utama, yaitu deteksi, pelacakan (*tracking*), normalisasi citra, dan regresi *landmark*, sebagaimana ditunjukkan pada Gambar 2.5 yang merupakan diagram *pipeline MediaPipe*.



Gambar 2. 2. Diagram *Pipeline Mediapipe* (F. Zhang dkk., 2020)

Proses dimulai dengan menerima input berupa *frame* video yang diambil secara kontinu dari kamera. Setiap *frame* berisi informasi visual yang akan dianalisis untuk mendeteksi keberadaan tangan, wajah, maupun pose tubuh. Pada tahap awal, *MediaPipe* menggunakan model deteksi berbasis *Single Shot Detector*

(SSD), yaitu *BlazePalm* untuk tangan, *BlazeFace* untuk wajah, dan *BlazePose* untuk pose tubuh, guna mengidentifikasi lokasi objek dalam *frame*.

Model deteksi ini bekerja menggunakan pendekatan *anchor-based*, di mana sejumlah *anchor boxes* dengan berbagai skala (misalnya 30×30 dan 10×10 *feature maps*) digunakan untuk mendeteksi objek dengan ukuran yang berbeda. Selain menghasilkan *bounding box*, model ini juga menghasilkan *orientation vector* yang menunjukkan arah orientasi objek, misalnya dari pergelangan tangan ke ujung jari. Informasi ini penting untuk menangani variasi rotasi objek pada citra (F. Zhang dkk., 2020).

Untuk meningkatkan efisiensi komputasi, *MediaPipe* menerapkan mekanisme *tracking*. Setelah objek berhasil dideteksi pada *frame* awal, posisi objek pada *frame* sebelumnya digunakan untuk menentukan *Region of Interest* (ROI) pada *frame* berikutnya. Dengan pendekatan ini, proses deteksi tidak perlu dijalankan kembali pada setiap *frame*, sehingga beban komputasi dapat dikurangi secara signifikan.

Setelah ROI diperoleh, dilakukan tahap normalisasi citra untuk meningkatkan konsistensi input ke model berikutnya. Tahap ini meliputi proses *cropping* pada area sekitar objek serta rotasi citra berdasarkan *orientation vector* agar objek memiliki orientasi yang seragam (misalnya pergelangan tangan berada di bawah dan jari mengarah ke atas). Normalisasi ini bertujuan untuk mengurangi variansi posisi dan rotasi sehingga meningkatkan akurasi model pada tahap selanjutnya.

Citra hasil normalisasi kemudian digunakan sebagai input untuk model kedua, yaitu landmark regressor berbasis arsitektur U-Net CNN, yang berfungsi sebagai encoder dan *decoder*. Bagian *encoder* (*down-sampling*) berfungsi untuk mengekstraksi fitur penting dari citra (representasi *what*), sedangkan bagian *decoder* (*up-sampling*) berfungsi untuk merekonstruksi informasi spasial guna menentukan posisi titik (*where*).

Untuk menjaga presisi lokasi *landmark*, digunakan mekanisme *skip connections* yang menghubungkan layer pada *encoder* ke *decoder*, sehingga informasi spasial beresolusi tinggi tidak hilang selama proses *down-sampling*.

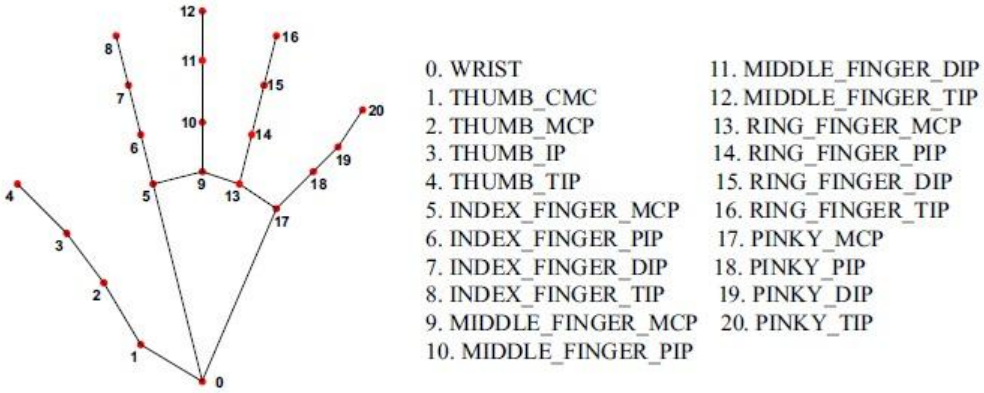
Model ini dilatih menggunakan kombinasi data nyata dan *synthetic* 3D data, sehingga mampu memperkirakan informasi kedalaman meskipun input berasal dari kamera 2D (F. Zhang dkk., 2020).

Output awal dari model ini berupa *probability heatmaps* untuk setiap titik *landmark*, yang kemudian dikonversi menjadi koordinat numerik. *MediaPipe* menghasilkan *landmark* dalam bentuk koordinat tiga dimensi, yaitu sumbu x, y dan z, dimana x dan y merupakan koordinat ter-normalisasi dalam rentang [0, 1], sedangkan *z* merepresentasikan kedalaman relatif terhadap titik referensi.

MediaPipe juga menyediakan fitur anotasi visual dengan menggambarkan titik-titik *landmark* beserta koneksinya pada *frame* video secara *real-time*. Ilustrasi keseluruhan landmark yang dihasilkan *MediaPipe* ditunjukkan pada Gambar 2.3, sedangkan detail *landmark* wajah dan tangan masing-masing ditunjukkan pada Gambar 2.4 dan Gambar 2.5.

Gambar 2. 3. Ilustrasi Keseluruhan *Landmark* dengan *MediaPipe* (Sumber: research.google/blog/mediaPipe-holistic-simultaneous-face-hand-and-pose-prediction-on-device/)

Gambar 2. 4. Titik *Face Landmark MediaPipe* (Sumber: *research.google/blog/ mediaPipe-holistic-simultaneous-face-hand-and-pose-prediction-on-device/*)



Gambar 2. 5. Titik *Hand Landmark MediaPipe* (Khartheesvar dkk., 2024)

MediaPipe dapat menghasilkan sampai dengan 543 *landmark* tubuh dengan koordinat x, y, dan z yang terstandarisasi untuk merepresentasikan gerakan tubuh, dan koordinat-koordinat ini sangat penting dalam menangkap gerakan dan isyarat yang dibuat oleh pengguna. (Khartheesvar dkk., 2024), Misalnya, jika kita menggunakan *MediaPipe* pada pose dalam mengenali gerakan tubuh, maka setiap *frame* akan memiliki 33 *landmark*, masing-masing dengan koordinat (x, y, z, *visibility*). Untuk sebuah video dengan 100 *frame* misalnya, data ini akan berbentuk *array* dengan dimensi (100, 33, 4). Jika menggunakan *MediaPipe* pada tangan, hasilnya akan memiliki 21 *landmark* pada satu tangan, sehingga menjadi 42 *landmark* pada kedua tangan, masing-masing dengan koordinat (x, y, z), sehingga bentuk *array*nya menjadi (100, 42, 3). *MediaPipe* juga mampu memberikan *landmark* yang konsisten dan akurat, yang selanjutnya meningkatkan performa model dalam mengenali gerakan isyarat dinamis dan statis. FPS (*Frame Per Second*) yang dihasilkan *MediaPipe* tidak bersifat tetap, melainkan tergantung pada berbagai faktor seperti kecepatan pemrosesan model, spesifikasi perangkat keras (CPU/GPU), serta resolusi video. Pada CPU, *MediaPipe* biasanya bekerja dengan kecepatan 10–30 FPS, sedangkan pada GPU bisa mencapai 30–60 FPS.

2.2.4. *Confusion Matrix*

Confusion matrix telah di gunakan terutama untuk mengukur dan mengevaluasi model. Matriks ini juga digunakan untuk menghitung sejumlah metrik kinerja model lainnya, seperti *precision* dan *recall*, di antara metrik lainnya. Struktur dari *confusion matrix* terdiri dari empat komponen utama, yaitu

True Positive (TP), yaitu jumlah yang benar positif dan diprediksi positif oleh model, *False positive* (FP), yaitu jumlah yang sebenarnya negatif tetapi diprediksi positif, *True Negative* (TN), yaitu jumlah yang benar negatif dan diprediksi negatif, dan *False negative* (FN), yaitu jumlah yang sebenarnya positif tetapi diprediksi negatif oleh model (Bora dkk., 2022). *Confusion matrix* biasanya akan divisualisasikan dalam bentuk tabel seperti pada Tabel 2.3, yang menunjukan jumlah dari setiap komponen yang ada.

Tabel 2. 3. *Confusion matrix (Hand & Christen, 2018)*

	Aktual	
	Positif	Negatif
Prediksi	TP	FN
	FP	TN

Metrik-Metrik yang dihitung dengan melihat ke struktur *confusion matrix* meliputi akurasi, *recall*, *precision*, dan *F1-score*, untuk melakukan evaluasi terhadap model yang telah dibangun, Akurasi adalah proporsi prediksi yang benar dari total prediksi, yang mana dirumuskan dalam Persamaan 2.11.

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.11}$$

Selanjutnya, presisi yang digunakan untuk mengukur sampel kelas positif yang diprediksi dengan benar, atau proporsi prediksi positif yang benar dari semua prediksi positif, yang mana dirumuskan dalam Persamaan 2.12 (Hand & Christen, 2018).

$$\text{Presisi} = \frac{TP}{TP+FP} \tag{2.12}$$

Dengan TP dan FP menunjukkan jumlah positif benar dan positif palsu, *recall* digunakan untuk menghitung semua sampel positif, perhitungan *recall* dirumuskan dalam Persamaan 2.13 (Hand & Christen, 2018).

$$Recall = \frac{TP}{TP+FN} \tag{2.13}$$

F1-score yang merupakan rata-rata harmonis dari *precision* dan *recall*, yang memberikan keseimbangan antara kedua metrik tersebut. *F1-score* memberikan skor antara 0 dan 1, di mana 1 menunjukkan model yang sempurna,

dan 0 menunjukkan kinerja yang buruk. *F1-score* sangat berguna saat ada ketidakseimbangan antara jumlah contoh positif dan negatif di dalam dataset. *F1-score* dirumuskan sebagai Persamaan 2.14 (Hand & Christen, 2018).

$$F1\text{-}score = 2 \times \frac{\text{Presisi} \times Recall}{\text{Presisi} + Recall} \quad (2.14)$$

2.2.5. *Robustness Testing*

Metrik lainnya juga di implementasikan dalam penelitian ini, yang mana untuk menilai kemampuan model dalam menangkap pola secara keseluruhan, Diantaranya seperti *Balanced Accuracy* untuk melihat keseimbangan prediksi, yang mana dirumuskan dengan Persamaan 2.15 (Sujon dkk., 2025).

$$\text{BA} = \frac{1}{2} \times \left(\frac{\text{TP}}{\text{TP}+\text{FN}} + \frac{\text{TN}}{\text{TN}+\text{FP}}\right) \quad (2.15)$$

ROC-AUC (*Area Under Curve of ROC*) yang digunakan untuk mengukur kemampuan model dalam membedakan kelas, dirumuskan dengan Persamaan 2.16 (Sujon dkk., 2025).

$$\text{AUC} = \int_0^1 TPR(FPR)d(FPR) \quad (2.16)$$

Selanjutnya, *Matthews correlation coefficient* (MCC), yang dianggap sebagai metrik paling komprehensif untuk masalah binary *clas*sification, terutama untuk data tidak seimbang, dirumuskan dengan Persamaan 2.17 (Sujon dkk., 2025).

$$\text{MCC} = \frac{(TP.TN - FP.FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (2.17)$$

Robustness testing lainnya juga dimplementasikan untuk mengevaluasi kekokohan model dalam skema dunia nyata, yaitu dengan memberikan *noise*, *frame masking*, dan *temporal rotation* terhadap data *valid*, sehingga membentuk 3 data *test* baru, lalu melihat *accuracy* yang dihasilkan terhadap 3 data *test* terbaru tersebut, Menambahkan *noise* pada data dirumuskan dengan Persamaan 2.18 (Sujon dkk., 2025).

$$\text{Ny} = y_t + N(0, \rho^2) \quad (2.18)$$

Ny menandakan sebuah data *test* baru, dengan menambahkan *noise* yang di tandai dengan ρ 0,5, kedalam dataset y_t dalam *temporal landmark* nya. Mensimulasikan *noise* sensor berfungsi untuk menguji apakah model tetap stabil

kalau *input* sedikit kotor. Selanjutnya, *Frame masking* juga dilakukan, untuk mensimulasikan *input* yang missing data seperti beberapa titik hilang, atau *frame* yang tidak baik, sehingga menguji model apakah masih bisa mengenali pola meskipun sebagian informasi hilang. *Frame masking* ini di tandai dengan Persamaan 2.19.

$$\text{MK y} = M_t . y_t = \int_{y_t,1-p}^{0,p} \tag{2.19}$$

MKy menandakan data *test* baru yang telah di*masking*, dimana M adalah *masking* ke-t, dengan p adalah probabilitas 0,03, artinya tiap elemen punya probabilitas 0,03 untuk di-set ke 0. Selanjutnya, *Temporal rotation* atau rotasi spasial 2D, yang mana dirumuskan dengan Persamaan 2.20.

$$\text{TPy} = R(\theta)P_t \tag{2.20}$$

TPy menandakan data *test* baru yang telah di lakukan *temporal rotation*, yang mana untuk setiap *frame* Pt, diambil setiap koordinat 2D, yang ditandai dengan Persamaan 2.21.

$$P_t = \frac{Xt}{Yt} \tag{2.21}$$

Setelahnya, dirotasikan *frame* tersebut dengan matriks rotasi, seperti dalam persamaan 2.22. dimana θ adalah *radian* 0,7.

$$R\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{2.22}$$