

**PENERAPAN TEKNIK AUGMENTASI GEOMETRIK DAN *EDGE
DETECTION* PADA *PRE-PROCESSING* DATASET BAHASA ISYARAT
INDONESIA (BISINDO) UNTUK MENGATASI VARIASI
GERAKAN *REAL-TIME* DALAM APLIKASI ELCUE**

TUGAS AKHIR DALAM BENTUK LAIN SETARA SKRIPSI

RANI WIDYA ASTUTI

211401018



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

**PENERAPAN TEKNIK AUGMENTASI GEOMETRIK DAN *EDGE
DETECTION* PADA *PRE-PROCESSING* DATASET BAHASA ISYARAT
INDONESIA (BISINDO) UNTUK MENGATASI VARIASI
GERAKAN *REAL-TIME* DALAM APLIKASI ELCUE**

TUGAS AKHIR DALAM BENTUK LAIN SETARA SKRIPSI

Diajukan untuk melengkapi tugas dan memenuhi syarat untuk memperoleh ijazah
Sarjana Ilmu Komputer

**RANI WIDYA ASTUTI
211401018**



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

PERSETUJUAN

Judul : PENERAPAN TEKNIK AUGMENTASI GEOMETRIK DAN *EDGE DETECTION* PADA *PRE PROCESSING* DATASET BAHASA ISYARAT INDONESIA (BISINDO) UNTUK MENGATASI VARIASI GERAKAN *REAL-TIME* DALAM APLIKASI ELCUE

Kategori : TUGAS AKHIR DALAM BENTUK LAIN SETARA SKRIPSI

Nama : RANI WIDYA ASTUTI

Nomor Induk Mahasiswa : 211401018

Program Studi : S-1 ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

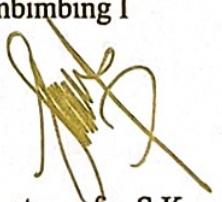
Komisi Pembimbing :
Medan, 27 Maret 2025

Pembimbing II



Fuzy Yustika Manik S.Kom., M.Kom
NIP. 198710152019032010

Pembimbing I



Hayatunnurus S.Kom, M.Cs
NIP. 199207192024062001

Diketahui/disetujui oleh

Program Studi S-1 Ilmu Komputer

Dr. Amalia S.T., M.T.

NIP. 197812212014042001

PERNYATAAN

PENERAPAN TEKNIK AUGMENTASI GEOMETRIK DAN *EDGE DETECTION* PADA *PRE-PROCESSING* DATASET BAHASA ISYARAT INDONESIA (BISINDO) UNTUK MENGATASI VARIASI GERAKAN *REAL-TIME* DALAM APLIKASI ELCUE

TUGAS AKHIR DALAM BENTUK LAIN SETARA SKRIPSI

Saya mengakui bahwa tugas akhir dalam bentuk lain setara skripsi ini merupakan hasil kerja sama tim dalam kompetisi. Penulisan judul dan isi dari tugas akhir dalam bentuk lain setara skripsi ini dilakukan secara individu dan berbeda satu dengan lainnya, yang disesuaikan dengan bidang masing-masing dalam tim, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 27 Maret 2025



Rani Widya Astuti

211401018

PENGHARGAAN

Segala puji dan syukur penulis ucapkan ke hadirat Allah Subhanahu Wa Ta’ala atas limpahan berkah, rezeki, rahmat serta ridho sehingga penulis dapat menyelesaikan bentuk lain setara skripsi ini untuk memenuhi salah satu syarat untuk lulus dan mendapat gelar Sarjana Komputer pada program studi S-1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.

Dengan penuh rasa hormat pada kesempatan ini penulis mengucapkan terima kasih kepada kedua Orang Tua penulis, Ayah, Ahmad Rivai Sihotang dan Ibu, Supiyah yang selalu memberikan dukungan tanpa batas, doa yang tulus, bantuan penuh kasih, serta dukungan materi yang tak ternilai, sehingga penulis dapat menjalani perkuliahan hingga menyelesaikan tugas akhir dalam bentuk lain setara skripsi ini tanpa beban. Penulis juga mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak atas segala dukungan, bantuan, serta doa yang diberikan kepada penulis selama menyelesaikan tugas akhir dalam bentuk lain setara skripsi ini. Adapun pada kesempatan kali ini penulis ingin mengucapkan rasa syukur dan terima kasih kepada:

1. Allah Subhanahu Wa Ta’ala yang senantiasa memberikan kemudahan, kekuatan, petunjuk dan ridho sehingga penulis dapat menyelesaikan bentuk lain setara skripsi ini dengan lancar dan penuh keberkahan.
2. Bapak Prof. Dr. Muryanto Amin S.Sos., M.Si. selaku Rektor Universitas Sumatera Utara.
3. Ibu Dr. Maya Silvi Lydia B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Bapak Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M. selaku Wakil Dekan I Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
5. Ibu Dr. Amalia S.T., M.T. selaku Ketua Program Studi S-1 Ilmu Komputer Universitas Sumatera Utara.
6. Ibu Hayatunnufus S.Kom, M.Cs. selaku Dosen Pembimbing I yang telah berjuang, mendukung, membimbing dengan sepenuh hati mulai dari awal

- terbentuknya ElCue hingga pada PIMNAS-37.
7. Ibu Fuzy Yustika Manik S.Kom., M.Kom selaku Dosen Pembimbing II sekaligus Dosen Penasehat Akademik yang telah membimbing, memberikan arahan, serta mendukung penulis baik dalam penyusunan bentuk lain setara skripsi ini maupun dalam perjalanan akademik secara keseluruhan.
 8. Bapak dan Ibu Dosen Program Studi S-1 Ilmu Komputer Universitas Sumatera Utara yang telah mengajar dan membagi ilmu, wawasan, serta pengalaman kepada penulis selama masa perkuliahan.
 9. Staf dan pegawai Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara yang siap sedia membantu urusan administrasi perkuliahan hingga dalam penyelesaian bentuk lain setara skripsi ini.
 10. Saudara kembar penulis, Ranti Widya Ningsih dan Abang penulis, Rebi Firmansyah Sihotang S.Agr. yang selalu menemani, mendukung kegiatan penulis, mendoakan dan sangat membantu penulis selama menjalani perkuliahan.
 11. Keluarga besar dari Ayah dan Ibu yang telah memberikan kasih sayang, doa, serta dukungan yang tulus dan tak ternilai sepanjang perjalanan ini.
 12. Sahabat penulis, Tessa Agitha Irwani Br. Barus, Putri Andriyani, Angela Siadari, Helga Priscilla Br. Purba, yang telah menjadi bagian dari perjalanan penulis, berbagi suka dan duka, serta berjuang bersama dalam setiap tantangan hingga PIMNAS-37. Terkhusus, Imanda Tamara Br. Pasaribu, yang telah setia menemani sejak awal perkuliahan, selalu memberikan dukungan, semangat, serta doa di setiap langkah perjalanan penulis. Setiap momen yang dilalui bersama adalah pengalaman berharga yang akan selalu penulis kenang.
 13. Teman-teman Cerita Kom B, Pieter, Vio, Lorenzo, Mutia, Elisa, Haikal, Harry, Reysha, Dita, Andhika dan lainnya. Serta teman-teman Angkatan 2021 yang telah memberikan peran selama masa perkuliahan di program studi Ilmu Komputer.
 14. Kakak Tingkat dan Adik Tingkat di Fasilkom-TI yang bersedia mendukung, bersama dalam kepanitiaan, serta memberi semangat kepada penulis.

Dan seluruh pihak yang telah memberikan dukungan serta doa baik yang tidak dapat penulis sebutkan satu per-satu. Penulis menyadari bahwa dalam penulisan bentuk lain setara skripsi ini masih terdapat kekurangan. Oleh karena itu, penulis mengharapkan adanya kritik dan saran yang membangun untuk penyempurnaan bentuk lain setara skripsi ini.

Medan, 27 Maret 2025

Penulis



Rani Widya Astuti
211401018



ABSTRAK

Kemajuan teknologi komunikasi, seperti *video call* telah mempermudah interaksi sosial, namun belum sepenuhnya inklusif bagi penyandang disabilitas Tuli. Bahasa Isyarat Indonesia (BISINDO) sebagai alat komunikasi utama bagi penyandang Tuli masih memiliki tantangan dalam penerapan deteksi otomatis, berupa variasi gerakan tangan, perbedaan sudut pandang, serta kondisi pencahayaan. Salah satu upaya adalah dengan menerapkan *pre-processing* dataset yang dapat meningkatkan variasi data dan *robustnes* model dalam mendeteksi gerakan selama panggilan video. Penelitian ini menerapkan *pre-processing* dataset menggunakan teknik Augmentasi Geometrik (rotasi, translasi, *scaling*, *shearing*, dan *flipping*) dan *Edge Detection* dengan *Gaussian Blur*, *Adaptive Thresholding*, dan *Otsu's* untuk memperbanyak variasi dataset sekaligus memperjelas kontur dan batas tangan. Hasil penelitian menunjukkan pertambahan dataset sebesar 380.800 gambar dengan tetap mempertahankan bentuk dasar isyarat tangan. Model *Convolutional Neural Network* (CNN) yang dilatih dengan dataset hasil *pre-processing* mencapai akurasi validasi sebesar 83.77% dan akurasi pengujian sebesar 83.88%, menunjukkan bahwa metode *pre-processing* ini efektif meningkatkan akurasi klasifikasi. Dengan demikian, penerapan *Edge Detection* dan Augmentasi Geometrik pada *pre-processing* BISINDO terbukti sangat berkontribusi dalam meningkatkan performa model, serta berpotensi untuk diterapkan dalam sistem pendekripsi bahasa isyarat *real-time* pada aplikasi *video call* ElCue.

Kata Kunci: *Pre-processing*, Augmentasi Geometrik, *Edge Detection*, BISINDO, *Real-time*

**IMPLEMENTATION OF GEOMETRIC AUGMENTATION AND EDGE
DETECTION IN PRE-PROCESSING INDONESIAN SIGN LANGUAGE
(BISINDO) DATASET TO HANDLE REAL-TIME MOVEMENT
VARIATIONS IN THE ELCUE APPLICATION**

ABSTRACT

Advancements in communication technology, such as video calls, have significantly improved social interactions. However, they remain inaccessible to individuals with hearing impairments. Indonesian Sign Language (BISINDO), the primary communication method for the Deaf community, presents challenges for automated recognition, particularly due to variations in hand gestures, differences in viewing angles, and inconsistent lighting conditions. One approach to addressing these challenges is by implementing pre-processing techniques to enhance data diversity and improve model robustness in gesture recognition during video calls. This research applies pre-processing techniques using Geometric Augmentation (rotation, translation, scaling, shearing, and flipping) and Edge Detection with Gaussian Blur, Adaptive Thresholding, and Otsu's Thresholding to increase dataset variation while simultaneously enhancing the clarity of hand contours and boundaries. The results show an expansion of the dataset to 380,800 images, preserving the fundamental structure of hand gestures. A Convolutional Neural Network (CNN) model trained on the pre-processed dataset achieved a validation accuracy of 83.77% and a test accuracy of 83.88%, confirming that the applied pre-processing methods significantly enhance classification accuracy. Consequently, the integration of Edge Detection and Geometric Augmentation in BISINDO pre-processing plays a crucial role in improving model performance and holds strong potential for implementation in real-time sign language recognition systems within the ElCue video call application.

Keywords: Pre-processing, Geometric Augmentation, Edge Detection, BISINDO, Real-time

DAFTAR ISI

PERSETUJUAN.....	i
PERNYATAAN.....	ii
PENGHARGAAN.....	iii
ABSTRAK	vi
ABSTRACT	vii
DAFTAR ISI.....	viii
DAFTAR TABEL.....	x
DAFTAR GAMBAR.....	xi
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah.....	3
1.3. Tujuan Penelitian.....	3
1.4. Batasan Penelitian.....	3
1.5. Manfaat Penelitian.....	4
1.6. Metodologi Penelitian	4
1.6.1. Studi Literatur	4
1.6.2. Analisis Permasalahan	4
1.6.3. Perancangan Sistem	5
1.6.4. Implementasi.....	5
1.6.5. Pengujian Sistem.....	5
1.6.6. Penyusunan Laporan	5
1.7. Sistematika Penulisan	6
BAB 2 LANDASAN TEORI	7
2.1. Bahasa Isyarat Indonesia (BISINDO).....	7
2.2. Computer Vision.....	7
2.3. Pre-processing Data.....	8
2.4. Augmentasi Data	8
2.4.1. Transformasi Geometrik	9
2.5. Edge Detection.....	11
2.5.1. Gaussian Blur.....	12

2.5.2. <i>Adaptive Thresholding</i>	13
2.5.3. <i>Otsu Thresholding</i>	13
2.6. <i>Convolutional Neural Network (CNN)</i>	14
2.7. Penelitian Terkait.....	16
BAB 3 ANALISIS DAN PERANCANGAN.....	20
3.1. Analisis Sistem.....	20
3.1.1. Analisis Masalah	20
3.1.2. Analisis Data	22
3.1.3. Analisis Kebutuhan	23
3.1.4. Arsitektur Umum Sistem.....	24
3.2. Pemodelan Sistem.....	26
3.2.1. <i>Activity Diagram</i>	26
3.2.2. <i>Flowchart</i>	27
BAB 4 IMPLEMENTASI DAN PENGUJIAN.....	29
4.1. Implementasi Sistem	29
4.1.1. <i>Pre-processing</i> Data.....	29
4.1.2. Penerapan <i>Edge Detection</i>	32
4.1.3. Penerapan Transformasi Geometrik.....	34
4.1.4. Model CNN untuk Pendeksiian.....	36
4.2. Pengujian Sistem	40
4.2.1. Pengujian Dataset Hasil <i>Pre-processing</i>	40
4.2.2. Pengujian Kinerja Model CNN	45
BAB 5 KESIMPULAN DAN SARAN	49
5.1. Kesimpulan	49
5.2. Saran	49
DAFTAR PUSTAKA	51
LAMPIRAN.....	54

DAFTAR TABEL

Tabel 2.1. Penelitian Terkait	18
Tabel 3.1. Kebutuhan Perangkat Lunak	24
Tabel 3.2. Kebutuhan Perangkat Keras	24
Tabel 4.1. Efek Augmentasi Geometrik pada Gambar.....	44



DAFTAR GAMBAR

Gambar 2.1. Komponen Model <i>Convolutional Neural Network</i>	15
Gambar 3.1. Diagram Ishikawa.....	22
Gambar 3.1. Arsitektur Umum Aplikasi <i>Video call ElCue</i>	25
Gambar 3.2. Arsitektur Umum Sistem <i>Pre-processing</i>	25
Gambar 3.3. <i>Activity Diagram</i>	27
Gambar 3.4. <i>Flowchart</i> Penelitian.....	28
Gambar 4.1. Dataset BISINDO dari Platform Kaggle	30
Gambar 4.2. Program Konfigurasi Kaggle.....	30
Gambar 4.3. Program Unduh Dataset.....	31
Gambar 4.4. Program Impor <i>Library</i>	31
Gambar 4.5. Program Memuat Dataset	32
Gambar 4.6. Program Penerapan <i>Edge Detection</i>	33
Gambar 4.7. Program <i>Load Data</i>	34
Gambar 4.8. Program Mengacak Data Latih dan Uji.....	34
Gambar 4.9. Program Penerapan Augmentasi.....	35
Gambar 4.10. Program Arsitektur Model CNN	37
Gambar 4.11. Program Kompilasi Model CNN	38
Gambar 4.12. Program <i>Training</i> Model CNN.....	38
Gambar 4.13. Proses <i>Training</i> Model CNN	39
Gambar 4.14. Program Visualisasi Data Asli	41
Gambar 4.15. Visualisasi Dataset Asli	41
Gambar 4.16. Program Visualisasi Hasil <i>Edge Detection</i>	41
Gambar 4.17. Visualisasi Hasil <i>Edge Detection</i>	42
Gambar 4.18. Program Visualisasi Hasil Augmentasi Geometrik	43
Gambar 4.19. Visualisasi Hasil Augmentasi Geometrik	44
Gambar 4.20. Program Uji Model CNN.....	46
Gambar 4.21. Grafik <i>Training</i> dan <i>Validation Accuracy</i>	46
Gambar 4.22. Grafik <i>Training</i> dan <i>Validation Loss</i>	47
Gambar 4.23. Program Evaluasi Model CNN	48
Gambar 4.24. Hasil Evaluasi Model CNN	48

BAB 1

PENDAHULUAN

1.1. Latar Belakang

Kemajuan teknologi saat ini telah berkembang pesat dan minat terhadap penggunaan teknologi sebagai alat bantu semakin meningkat (Korn dkk., 2018). Penggunaan teknologi komunikasi terkini, seperti *video call* sering menjadi opsi utama dalam berinteraksi sosial (Lubis dan Sani, 2014). *Video call* mempermudah individu untuk berkomunikasi secara langsung melalui telepon, inovasi canggih ini membuat kendala jarak jauh tidak lagi menjadi hambatan (Pratiwi, 2017). Namun kemajuan teknologi tersebut belum dapat dirasakan oleh semua masyarakat, salah satunya ialah penyandang disabilitas Tuli. Menurut data Badan Pusat Statistik Nasional tahun 2022, jumlah kesulitan mendengar di Indonesia mencapai 4.956.814 dari total penduduk sekitar 275.500.000 jiwa. Dimana jumlah penduduk yang sama sekali tidak bisa mendengar mencapai 255.161 jiwa. Hal ini menunjukkan potensi kurangnya interaksi sosial bagi mereka.

Sistem Bahasa Isyarat Indonesia (SIBI) telah diterapkan sebagai bahasa isyarat resmi di Sekolah Luar Biasa (SLB) secara resmi pada tahun 1996. Namun, karena Bahasa Isyarat Indonesia (BISINDO) merupakan bahasa yang berkembang dari bahasa ibu, masyarakat Tuli lebih cenderung menggunakan dalam kehidupan sehari-hari (Rizky dkk., 2023). Kurangnya pemahaman mengenai komunikasi menggunakan bahasa isyarat, khususnya BISINDO, di kalangan masyarakat non-Tuli mengakibatkan rendahnya inklusivitas komunikasi antara kelompok Tuli dan non-Tuli. Ketidakmampuan berkomunikasi dengan efektif sering kali menimbulkan kesalahpahaman, kebingungan, bahkan isolasi sosial bagi individu Tuli.

Untuk mengatasi kesenjangan teknologi dan meningkatkan inklusivitas, aplikasi ElCue hadir untuk memungkinkan komunikasi lebih efektif melalui *video call*. Teknologi yang mendukung pendekripsi dalam *video call* tersebut adalah *Computer Vision* dan *Deep Learning*. *Computer Vision* dapat diartikan sebagai sistem yang memungkinkan komputer untuk "melihat" dan menafsirkan informasi visual (misalnya, gambar atau video) seperti yang dilakukan manusia melalui

implementasi perangkat lunak atau perangkat keras (Normalisa dkk., 2022). Dengan memanfaatkan teknologi *Computer Vision*, aplikasi dapat mendeteksi dan menginterpretasikan gerakan Bahasa Isyarat Indonesia (BISINDO) menggunakan kamera seluler. Teknologi *Deep Learning* dibutuhkan untuk memungkinkan sistem secara otomatis memperbarui model pendekripsi agar dapat lebih akurat dalam memahami dan merespons variasi gerakan BISINDO seiring berjalannya waktu. *Deep Learning* sangat efektif untuk menangani data yang besar dan berdimensi tinggi karena kemampuannya mengekstraksi fitur kompleks secara otomatis melalui banyak lapisan pada jaringan neural (Janiesch dkk., 2021).

Kedua teknologi tersebut mampu mendekripsi gerakan isyarat dengan tingkat akurasi yang sangat baik, namun berbagai faktor, seperti perbedaan kecepatan gerakan, sudut pandang, dan kondisi pencahayaan, masih menjadi tantangan besar ketika diterapkan dalam aplikasi *real-time*, yaitu *video call*. Oleh karena itu, fokus penelitian ini adalah penerapan Data Augmentasi dan teknik *Edge Detection* sebagai solusi efektif untuk meningkatkan kinerja model dalam mendekripsi gerakan Bahasa Isyarat Indonesia (BISINDO).

Augmentasi data bertujuan untuk memperbanyak variasi data pelatihan melalui teknik-teknik tertentu, sehingga model dapat lebih baik mengenali pola gerakan dalam berbagai kondisi yang bervariasi. Augmentasi Data membuat model *Deep Learning* lebih tangguh terhadap berbagai jenis *noise*, serta meningkatkan jumlah data pelatihan, yang umumnya diperlukan dalam berbagai aplikasi (Elgendi dkk., 2021). Sementara itu, *Edge Detection* digunakan untuk memperjelas batas-batas gerakan, meningkatkan akurasi deteksi, serta membantu sistem mendekripsi variasi gerakan yang mungkin terjadi dalam situasi dunia nyata. Kombinasi kedua teknik ini diharapkan dapat mengatasi tantangan variasi gerakan dalam komunikasi menggunakan bahasa isyarat.

Berdasarkan penelitian sebelumnya yang dilakukan oleh Baiti-Ahmad Awaluddin, Chun-Tang Chao, dan Juing-Shian Chiou pada tahun 2023, dimana penelitian tersebut membahas pemanfaatan teknik augmentasi data berbasis transformasi geometrik untuk meningkatkan akurasi model dalam mengenali gerakan tangan statis. Penelitian tersebut menitikberatkan pada eksplorasi metode seperti rotasi, *scaling*, dan *flipping* untuk memperbanyak variasi data pelatihan,

sehingga model konvolusional yang telah dilatih sebelumnya dapat lebih optimal dalam mendeteksi pola gerakan tangan. Mengacu pada penelitian tersebut, penulis menggabungkan pendekatan Augmentasi Data dengan teknik *Edge Detection* untuk mengatasi tantangan pada aplikasi *real-time*. Penggabungan ini bertujuan tidak hanya untuk memperluas keragaman data, tetapi juga mengambil fitur penting seperti kontur dan batas gerakan, sehingga model dapat lebih *robust* dan akurat dalam mendeteksi gerakan Bahasa Isyarat Indonesia (BISINDO) dalam kondisi dinamis dan variatif selama panggilan video.

1.2. Rumusan Masalah

Terwujudnya inklusivitas antara penyandang disabilitas Tuli dan masyarakat non-Tuli dalam komunikasi melalui *video call* masih menghadapi berbagai tantangan, terutama dalam pendekripsi Bahasa Isyarat Indonesia (BISINDO) secara *real-time*. Variasi dalam gerakan tangan, kecepatan, dan ekspresi menyebabkan sistem pendekripsi kesulitan dalam mengenali isyarat secara akurat. Kualitas data yang dipengaruhi oleh *noise*, warna, dan tekstur yang tidak seragam menjadi kendala dalam ekstraksi fitur secara optimal. Selain itu, transformasi data yang kurang variatif dalam posisi dan skala tangan juga mempengaruhi akurasi sistem. Tantangan ini menunjukkan bahwa sistem deteksi BISINDO dalam *video call real-time* ElCue masih belum optimal dalam mengatasi variasi bahasa isyarat, sehingga dapat menghambat komunikasi yang efektif antara penyandang Tuli dan non-Tuli.

1.3. Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah untuk menerapkan teknik Augmentasi Geometrik dan *Edge Detection* pada tahapan *pre-processing* dataset Bahasa Isyarat Indonesia (BISINDO) untuk menghasilkan dataset yang lebih banyak dan lebih jelas, sehingga model pendekripsi dapat mendekripsi variasi gerakan *real-time* yang dilakukan selama panggilan video dalam aplikasi ElCue.

1.4. Batasan Penelitian

Agar penelitian tidak meluas dan lebih terarah, dibutuhkan adanya batasan penelitian sebagaimana tertera pada poin-poin berikut:

1. Penelitian dibatasi pada tahap *pre-processing* dataset BISINDO.
2. Pelatihan dan pengujian dataset hasil *pre-processing* menggunakan model

Convolutional Neural Network.

3. Dataset yang digunakan terbatas pada 9 isyarat BISINDO, yaitu “A”, “Asal”, “B”, “C”, “D”, “E”, “Halo”, “I”, “Terima Kasih”.
4. Penelitian ini menggunakan bahasa pemrograman Python.
5. Teknik Augmentasi Data yang diterapkan adalah Transformasi Geometrik.
6. Teknik *Edge Detection* yang diterapkan adalah metode *Gaussian Blur*, *Threshold Adaptif* dan *Otsu Thresholding*, yang parameternya telah ditetapkan.

1.5. Manfaat Penelitian

Adapun manfaat dari penelitian ini, yaitu:

1. Meningkatkan akurasi dan *robustnes* pendekripsi gerakan isyarat BISINDO dalam aplikasi ElCue melalui penerapan teknik Augmentasi Geometrik dan *Edge Detection*, sehingga *output* pendekripsi *real-time* dapat tervisualisasikan dengan jelas.
2. Meningkatkan kemampuan sistem dalam mengatasi variasi gerakan *real-time* yang terjadi selama panggilan video di aplikasi ElCue.
3. Mendukung pengembangan sistem pendekripsi Bahasa Isyarat yang lebih adaptif dan efektif, untuk menciptakan komunikasi yang inklusif antara penyandang disabilitas Tuli dan masyarakat non-Tuli.

1.6. Metodologi Penelitian

Adapun langkah – langkah yang dilakukan dalam penelitian ini adalah sebagai berikut:

1.6.1. Studi Literatur

Sebagai tahap awal, dilakukan kajian pustaka yang komprehensif dengan merujuk pada berbagai publikasi ilmiah, baik berupa buku, jurnal, makalah, serta berbagai sumber lainnya yang memiliki keterkaitan dengan penelitian penulis, di antaranya penelitian mengenai penerapan teknik Augmentasi Geometrik dan *Edge Detection* pada *pre-processing* dataset bahasa isyarat.

1.6.2. Analisis Permasalahan

Pada tahapan ini, penulis melakukan analisis referensi dan informasi yang

sebelumnya telah dikumpulkan untuk mendapat pemahaman tentang penerapan teknik Augmentasi Geometrik dan penerapan *Edge Detection* sebagai metode pada tahap *pre-processing* dataset yang dapat meningkatkan akurasi dan *robustness* pendekripsi gerakan isyarat secara *real-time* dalam panggilan video.

1.6.3. Perancangan Sistem

Pada tahap ini, penulis merancang arsitektur umum yang menjadi kerangka kerja dari program yang akan dikembangkan. Fokus perancangan diarahkan pada pembuatan program *pre-processing* yang mengintegrasikan teknik Augmentasi Geometrik dan *Edge Detection* untuk mengolah dataset Bahasa Isyarat Indonesia (BISINDO) dan mengujinya menggunakan model *Convolutional Neural Network* (CNN).

1.6.4. Implementasi

Pada tahap implementasi, rancangan yang telah disusun diubah menjadi kode program yang fungsional. Penulis mengembangkan program *pre-processing* yang menerapkan teknik Augmentasi Geometrik, seperti rotasi, *scaling*, *flipping*, *shearing* dan translasi untuk memperbanyak variasi data pelatihan, serta menerapkan metode *Edge Detection* menggunakan *Threshold Adaptif* dan *Otsu Thresholding* untuk menyoroti kontur gerakan isyarat secara optimal.

1.6.5. Pengujian Sistem

Tahap pengujian sistem dilakukan untuk mengevaluasi kualitas data setelah melalui tahap *pre-processing*. Proses ini memastikan bahwa data hasil *pre-processing*, yang mencakup Augmentasi Geometrik dan *Edge Detection*, telah menghasilkan dataset yang banyak, bervariasi serta berkонтur jelas. Setelah itu, pengujian dilakukan dengan menguji hasil *pre-processing* menggunakan model *Convolutional Neural Network* (CNN). Pengujian ini bertujuan untuk menilai apakah teknik *pre-processing* tersebut berhasil meningkatkan akurasi dan apakah model CNN dapat bekerja dengan optimal dalam kondisi variasi gerakan tangan.

1.6.6. Penyusunan Laporan

Pada tahap ini dilakukan penyusunan laporan untuk mempresentasikan hasil dari penelitian yang dilakukan.

1.7. Sistematika Penulisan

Adapun sistematika penulisan pada penelitian ini terdiri atas lima bab, yaitu sebagai berikut:

Bab 1: Pendahuluan

Bagian ini berisi latar belakang, rumusan masalah, tujuan penelitian, batasan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan dari penelitian yang dilakukan penulis.

Bab 2: Landasan Teori

Bagian ini berisikan teori yang menjadi dasar penelitian serta berkaitan dengan masalah yang ada dalam penelitian. Beberapa teori diantaranya adalah teori BISINDO, *Computer Vision*, *Pre-processing* Data, Augmentasi Data dan Transformasi Geometrik, *Edge Detection*, *Convolutional Neural Network* (CNN) serta Penelitian Terdahulu.

Bab 3: Analisis dan Perancangan

Bagian ini menjelaskan analisis sistem secara keseluruhan hingga perancangan arsitektur sistem. Sistem yang dirancang mencakup tahapan *pre-processing*, mulai dari dataset BISINDO yang digunakan, penerapan teknik Augmentasi dan penerapan *Edge Detection*. Hasil dari proses *pre-processing* ini kemudian diuji kualitasnya menggunakan model CNN, sehingga alur kerja sistem secara menyeluruh dapat dipastikan menghasilkan data yang optimal untuk pendekripsi isyarat dalam aplikasi ElCue.

Bab 4: Implementasi Dan Pengujian Sistem

Bagian ini membahas implementasi sistem berdasarkan perancangan yang telah disusun pada bab sebelumnya. Dalam bab ini, penulis menjelaskan bagaimana program *pre-processing* yang menggabungkan teknik Augmentasi Geometrik dan *Edge Detection*. Selanjutnya, bab ini menyajikan pengujian menggunakan model CNN untuk mengevaluasi kualitas data hasil *pre-processing*.

Bab 5: Kesimpulan dan Saran

Bagian ini berisi kesimpulan yang dapat diperoleh berdasarkan pemaparan pada setiap bab serta saran yang diberikan penulis sebagai masukan untuk penelitian kedepannya.

BAB 2

LANDASAN TEORI

2.1. Bahasa Isyarat Indonesia (BISINDO)

Bahasa Isyarat Indonesia atau BISINDO, merupakan bahasa isyarat yang tumbuh dan berkembang secara alami dalam disabilitas Tuli. BISINDO memiliki dua kategori isyarat tangan, yakni isyarat statis dan dinamis. Dalam penggunaannya, isyarat tangan dalam BISINDO mencakup abjad, angka, dan kata-kata yang secara langsung mengacu pada konsep atau objek tertentu (Hikmatia dan Zul, 2021). Bahasa Isyarat Indonesia (BISINDO) digunakan sebagai media komunikasi sehari-hari oleh disabilitas Tuli, yang mencakup penggunaan gerakan kedua tangan dan ekspresi wajah. Seperti halnya bahasa daerah, BISINDO memiliki variasi dalam penggunaannya di setiap daerah. Berbeda dengan Sistem Bahasa Isyarat Indonesia (SIBI), yang mengandalkan pengejaan sesuai dengan prinsip Ejaan Bahasa Indonesia (EBI), BISINDO tidak mengikuti kaidah EBI dan lebih mementingkan kesederhanaan serta ekspresivitas untuk menggambarkan peristiwa yang sedang terjadi (Fadillah dkk., 2021).

2.2. Computer Vision

Computer Vision merupakan cabang dari kecerdasan buatan (AI) yang berfokus pada bagaimana komputer dapat memperoleh, mengolah, dan memahami informasi visual dari dunia nyata melalui gambar atau video digital. *Computer Vision* sebagai salah satu bidang multidisiplin yang mempelajari cara komputer dalam menganalisis serta menafsirkan gambar dan video digital untuk menggantikan mata manusia untuk tujuan seperti pengenalan, penelusuran, pengukuran, dan masalah visual lainnya (Dong dkk., 2021). Informasi visual serta kemampuan *Computer Vision* sangat penting dalam memproses, menganalisis, dan memahami visual sebagai gambar atau video digital untuk menghasilkan keputusan yang bermakna dan mengambil tindakan (Nalbant dan Uyanik, 2021).

Dalam konteks penelitian ini, *Computer Vision* diterapkan untuk meningkatkan proses *pre-processing* dataset Bahasa Isyarat Indonesia (BISINDO) melalui integrasi teknik Augmentasi Geometrik dan *Edge Detection*. Dengan

menerapkan teknik-teknik tersebut, sistem diharapkan mampu menonjolkan fitur-fitur visual yang penting dan menghasilkan data pelatihan yang lebih kaya serta bervariasi.

2.3. *Pre-processing* Data

Data tidak selalu berada dalam kondisi "bersih", sehingga keberadaan data yang *redundant*, tidak konsisten, atau hilang dalam sebuah dataset menandakan bahwa data tersebut perlu ditangani terlebih dahulu sebelum diterapkan pada algoritma. *Pre-processing* data bertujuan untuk mengatasi masalah-masalah tersebut. Secara khusus, kinerja algoritma sangat dipengaruhi oleh teknik-teknik *pre-processing* yang diterapkan. Beberapa peneliti berpendapat bahwa penerapan teknik *pre-processing* tertentu sangat bergantung pada karakteristik dataset yang digunakan (Alshdaifat dkk., 2021). Dengan menerapkan proses *pre-processing* yang tepat, dataset menjadi lebih bersih dan terstruktur, sehingga algoritma dapat dilatih dengan data yang lebih akurat. Hal ini secara langsung meningkatkan performa dan keandalan sistem, karena kualitas *input* data merupakan fondasi utama yang menentukan hasil dari setiap pemodelan.

Secara umum, tahapan *pre-processing* data mencakup beberapa langkah utama. Pertama, data dikumpulkan dan dibersihkan dari *noise*, duplikasi, atau nilai yang hilang, sehingga menghasilkan dataset yang konsisten dan berkualitas. Selanjutnya, data dinormalisasi atau distandardisasi untuk memastikan skala yang seragam, lalu dilakukan transformasi seperti *feature engineering* atau *data augmentation* (pada kasus citra) untuk memperkaya variasi data. Terakhir, dataset dibagi menjadi subset untuk pelatihan, validasi, dan pengujian. Dalam penelitian ini, tahap *pre-processing* menjadi fondasi penting yang harus dilakukan sebelum dataset BISINDO diterapkan pada algoritma pendekripsi, untuk memastikan sistem dapat menghasilkan *output* yang optimal dalam kondisi *real-time*.

2.4. Augmentasi Data

Augmentasi data adalah teknik yang digunakan untuk memperbanyak dan memperkaya variasi dataset dengan cara mensintesis data baru dari data yang sudah ada. Teknik ini sangat penting terutama ketika jumlah data yang tersedia terbatas atau ketika data tersebut tidak mencakup semua variasi kondisi yang mungkin

terjadi dalam situasi nyata. Untuk membuat model *Deep Learning* yang dapat mengenali data dengan baik, para peneliti sering kali menghadapi kendala karena jumlah data yang tersedia terbatas. Untuk mengatasi masalah ini, mereka menggunakan teknik seperti augmentasi, *dropout*, dan *transfer learning* guna menambah jumlah data. Augmentasi data melibatkan pembuatan data tambahan secara otomatis selama proses pelatihan, sehingga model dapat belajar lebih efektif dan mengurangi kesalahan (Maharana dkk., 2022). Augmentasi data bertujuan untuk membuat model *Deep Learning* lebih tangguh terhadap berbagai jenis *noise* serta meningkatkan jumlah data pelatihan.

Di bidang pengolahan citra, langkah augmentasi (yaitu, menambah jumlah data pelatihan) sering digunakan untuk mengurangi *overfitting* pada dataset pelatihan serta meningkatkan akurasi prediksi pada dataset pengujian (Elgendi dkk., 2021). Dengan melakukan augmentasi, model dapat belajar dari berbagai skenario yang mungkin tidak tercakup dalam data asli, sehingga menjadi lebih *robust* saat diterapkan pada data baru. Contoh metode *data augmentation* meliputi: transformasi geometrik (rotasi, *flipping*, *scaling*, translasi, *cropping*), transformasi warna (penyesuaian kecerahan, kontras, saturasi, hue), penambahan *noise* (*noise Gaussian*, *salt and pepper noise*), serta teknik lain seperti *Gaussian blur*, *median filtering*, *Cutout*, dan *Mixup*. Secara keseluruhan, penggunaan augmentasi data di tahap *pre-processing* mempersiapkan data agar lebih *robust*, meningkatkan kualitas pelatihan, dan mendukung performa model yang lebih optimal ketika dihadapkan dengan data baru.

2.4.1. Transformasi Geometrik

Transformasi geometrik mencakup perubahan ukuran (*scaling*), memutar (rotasi), menggeser (translasi), mengubah bentuk (*shearing*), dan membalik gambar (*flipping*), dan merupakan salah satu cara yang paling umum digunakan dalam pengolahan citra. Teknik-teknik ini membantu menambah jumlah data yang tersedia untuk pelatihan model pembelajaran mendalam, sehingga dataset menjadi lebih seimbang dan proses pelatihan menjadi lebih efisien.

Penggunaan transformasi geometrik sudah banyak diterapkan dalam penelitian pengenalan gerakan tangan untuk memperkaya dataset secara efektif. Transformasi geometrik merupakan teknik yang sederhana namun sangat efektif,

karena mengubah gambar asli melalui operasi matematika dasar. Berdasarkan penelitian Awaluddin dkk., 2023, berikut adalah teknik-teknik dalam transformasi geometrik:

A. *Image Scaling*

Scaling gambar mengubah ukuran gambar *input* menjadi lebih besar atau lebih kecil dengan menerapkan faktor skala tertentu. Jika gambar diperbesar (*scaled up*), bagian yang melebihi ukuran aslinya dipotong agar sesuai, sedangkan jika gambar diperkecil (*scaled down*), ukuran aslinya tetap dipertahankan dengan ruang yang kosong diisi menggunakan metode tetangga piksel terdekat. Proses ini menggunakan persamaan (1) dan (2), di mana (x, y) adalah koordinat piksel pada gambar asli, (x', y') adalah koordinat pada gambar yang telah diskalakan, dan s_x serta s_y adalah faktor skala untuk baris dan kolom.

$$x' = x \cdot s_x \dots (1)$$

$$y' = y \cdot s_y \dots (2)$$

B. *Image Rotation*

Rotasi gambar melibatkan pemutaran gambar dengan sudut tertentu (biasanya antara 0 hingga 360 derajat) untuk menghasilkan data pelatihan tambahan. Persamaan (3) dan (4) digunakan untuk mengubah koordinat piksel asli (x, y) menjadi koordinat (x', y') pada gambar yang telah diputar, di mana sudut rotasi (dalam radian) dan koordinat pusat gambar (cx, cy) digunakan sebagai parameter.

$$x' = (x - cx)\cos\theta - (y - cy)\sin\theta + cx \dots (3)$$

$$y' = (x - cx)\sin\theta - (y - cy)\cos\theta + cy \dots (4)$$

C. *Image Translation*

Translasi gambar merupakan teknik yang menggeser gambar secara horizontal (sumbu x) dan vertikal (sumbu y) dengan sejumlah pergeseran piksel tertentu untuk menghasilkan data pelatihan tambahan. Teknik ini meningkatkan ketangguhan model terhadap variasi posisi objek dalam gambar. Persamaan (5) dan (6) digunakan untuk melakukan translasi pada sumbu x dan sumbu y secara berurutan, di mana (x, y) adalah koordinat piksel pada gambar asli, (x', y') adalah koordinat pada gambar yang telah

digeser, dan (dx, dy) merupakan nilai offset translasi.

$$x' = x + dx \dots (5)$$

$$y' = y + dy \dots (6)$$

D. Image Shearing

Shearing adalah teknik yang memiringkan gambar dengan menggeser setiap baris atau kolom piksel berdasarkan koordinatnya. Teknik ini memodifikasi posisi piksel sehingga gambar tampak miring, yang sangat berguna untuk menangani gambar yang diambil dari berbagai sudut atau perspektif. Proses *shearing* dapat dilakukan sepanjang sumbu x dan y menggunakan persamaan (7) dan (8) yang memetakan koordinat asli (x, y) menjadi koordinat baru (x', y') pada gambar yang telah di-*shear*, dengan faktor shear sh_x dan sh_y yang menentukan besarnya pergeseran pada masing-masing sumbu.

$$x' = x + sh_x \times y \dots (7)$$

$$y' = y + sh_y \times x \dots (8)$$

E. Image Flipping

Flipping merupakan proses yang membalik sisi kiri dan kanan dari sebuah gambar. Dalam pengenalan gerakan tangan, yang digunakan hanya *flipping* horizontal. Persamaan (9) menunjukkan rumus *flipping* horizontal, di mana (x, y) adalah koordinat piksel pada gambar asli, x' adalah indeks piksel yang bersesuaian pada gambar yang telah dibalik, dan W adalah lebar gambar.

$$x' = (W - 1) - x \dots (9)$$

2.5. Edge Detection

Edge Detection adalah salah satu masalah dasar dalam *Computer Vision* yang memiliki beragam aplikasi, seperti segmentasi gambar, pendekripsi objek, dan segmentasi objek dalam video. Tujuan dari *Edge Detection* adalah untuk mengambil batas objek yang akurat dan menonjolkan tepi-tepi penting pada gambar. Proses ini cukup menantang karena banyak faktor, seperti latar belakang yang kompleks dan anotasi yang tidak konsisten. Karena *Edge Detection* sangat bergantung pada konteks dan petunjuk semantik dalam gambar, sangat penting untuk mendapatkan representasi yang tepat agar dapat menangkap informasi visual tingkat tinggi maupun tingkat rendah secara efektif (Pu dkk., 2022). *Edge Detection*

terdahulu, seperti Sobel dan Canny, bekerja dengan menganalisis perubahan intensitas (gradien) pada gambar untuk menemukan tepi. Metode ini memberikan informasi dasar yang digunakan secara luas dalam berbagai aplikasi *Computer Vision* (Winnemoller dkk., 2012).

Tujuan utama dari *Edge Detection* adalah untuk mengidentifikasi perubahan kecerahan yang tajam pada gambar, seperti adanya diskontinuitas dalam intensitas, warna, atau tekstur. Berkat kemampuan *deep CNN* untuk secara otomatis mempelajari representasi data yang kaya dengan tingkat abstraksi yang berbeda, model-model ini telah membawa kemajuan besar dalam berbagai tugas *Computer Vision*, termasuk *Edge Detection*, dan terus berkembang pesat. Model *Edge Detection* berbasis *Deep Learning* awalnya membangun arsitektur CNN sebagai classifier yang memprediksi kemungkinan adanya *edge* pada potongan gambar (Su dkk., 2021).

2.5.1. *Gaussian Blur*

Gaussian blur adalah teknik pengaburan gambar yang menggunakan fungsi Gaussian untuk mengurangi *noise* dan *detail* halus yang tidak diinginkan. Karena kesederhanaan matematis distribusi Gaussian, keberadaannya yang melimpah dalam pustaka pengolahan citra, serta kesamaan kualitatifnya dengan fungsi penyebaran titik optik, seringkali lebih praktis menggunakan kernel Gaussian untuk mengaburkan gambar. Misalnya, berbagai studi tentang hubungan antara kualitas gambar dan *Deep Learning* telah memanfaatkan *Gaussian blur* untuk memanipulasi ketajaman gambar dan konten frekuensi spasial (Bergstrom dkk., 2023). Fungsi ini digunakan untuk mengurangi *noise* pada gambar dengan menerapkan fungsi Gaussian guna mengaburkan gambar. Dengan cara mengaburkan, *detail* yang tidak penting dihilangkan sehingga *noise* berkurang, sementara informasi penting dengan frekuensi spasial rendah tetap terjaga.

Persamaan (10) menggambarkan rumus *Gaussian blur*, di mana x menunjukkan arah pengukuran dan σ adalah simpangan baku dari distribusi Gaussian. Metode dua langkah pada Gaussian filtering ini membutuhkan lebih sedikit perhitungan, sehingga lebih efisien untuk diimplementasikan (Devi dkk., 2023).

$$g(x) = \frac{e^{-\frac{x^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}} \cdots (10)$$

2.5.2. Adaptive Thresholding

Dalam pengolahan citra digital, *Thresholding* adalah salah satu metode paling sederhana untuk melakukan segmentasi gambar. Teknik ini membagi gambar menjadi kelompok piksel berdasarkan nilai intensitasnya: piksel dengan nilai di bawah atau di atas ambang tertentu akan dikelompokkan secara terpisah. Ada beberapa teknik *Thresholding*, seperti metode global yang menggunakan histogram level abu-abu, metode global yang memperhitungkan properti lokal, serta *Adaptive Thresholding*. *Thresholding* global mengasumsikan bahwa histogram gambar memiliki dua puncak yang jelas (bimodal), di mana salah satunya mewakili objek dan yang lainnya mewakili latar belakang. Dalam hal ini, nilai ambang (*Thresholding*) biasanya dipilih sebagai titik minimum antara kedua puncak tersebut.

Untuk mengatasi keterbatasan *Thresholding* global, *Adaptive Thresholding* digunakan untuk menentukan nilai ambang secara lokal, baik dengan membagi gambar menjadi beberapa sub-gambar dan menghitung ambang untuk tiap bagian, maupun dengan mengukur intensitas di sekitar tiap piksel dan memilih ambang berdasarkan statistik seperti rata-rata lokal. Meskipun *Adaptive Thresholding* memerlukan perhitungan yang lebih kompleks, Teknik ini sangat berguna untuk mengatasi variasi intensitas pada gambar, terutama ketika latar belakang tidak seragam atau objek yang ingin dideteksi sangat kecil atau jarang (Jardim dkk., 2023).

2.5.3. Otsu Thresholding

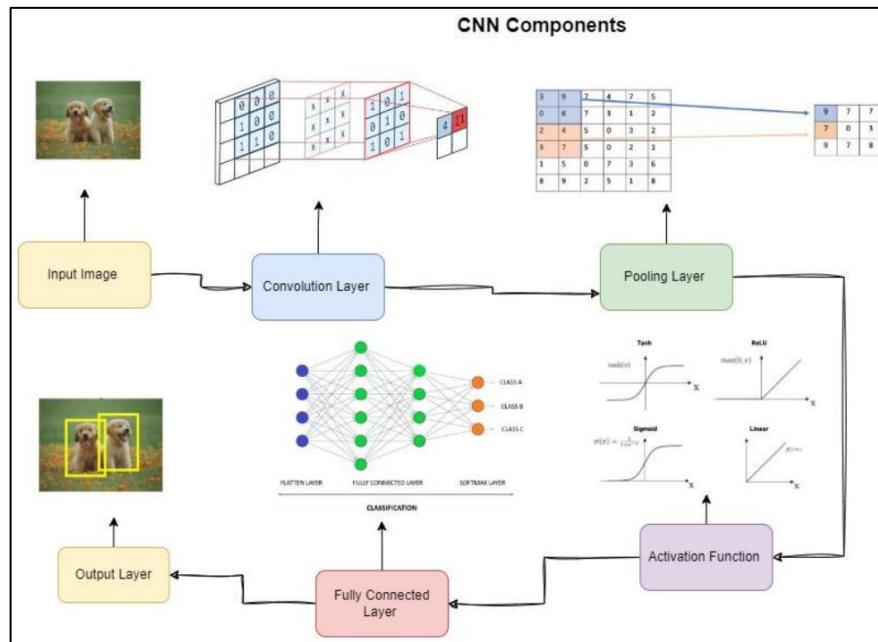
Salah satu metode global yang populer adalah metode Otsu, yang secara otomatis menentukan nilai ambang dengan memaksimalkan perbedaan antar kelas (Jardim dkk., 2023). *Thresholding* multi-level, seperti metode Otsu, merupakan salah satu metode *Thresholding* yang paling umum dalam pengolahan citra. Meskipun metode ini akurat dan efisien, metode tersebut memiliki kompleksitas komputasi yang tinggi; semakin banyak *Threshold* yang digunakan, maka efisiensi menurun karena kompleksitas dan waktu eksekusi yang meningkat. *Threshold* Otsu dapat

memberikan analisis gambar yang akurat dengan memilih *Threshold* optimal, yaitu titik di mana perbedaan variansi intensitas cahaya antara kedua sisi *Threshold* adalah maksimum.

Metode *Otsu Thresholding* bekerja dengan menggunakan informasi dari histogram gambar untuk menentukan *Threshold* optimal. Informasi histogram digunakan untuk menilai intensitas cahaya dalam gambar, yang kemudian dapat dianalisis dan dievaluasi. Misalnya, jika terdapat *Threshold* t, maka histogram akan dibagi menjadi dua kelas: kelas C0 yang mencakup piksel dengan intensitas antara 0 dan t-1, dan kelas C1 yang mencakup piksel dengan intensitas antara t dan L-1. Untuk gambar grayscale, diasumsikan L = 256 sebagai rentang kecerahan gambar (Al-rahlawee dan Rahebi, 2021). Metode Otsu berusaha memaksimalkan nilai target dengan memilih *Threshold* yang tepat. Dalam hal ini, semua *Threshold* harus dipertimbangkan. Namun, jika jumlah *Threshold* terlalu banyak, waktu pemrosesan metode Otsu akan meningkat secara signifikan sehingga menjadi tidak dapat diterapkan.

2.6. *Convolutional Neural Network* (CNN)

Berdasarkan penelitian Taye (2023), *Convolutional Neural Network* (CNN) adalah sistem kecerdasan buatan yang menggunakan jaringan saraf berlapis untuk mengenali, mengklasifikasikan, dan mendeteksi objek dalam gambar. Berbeda dengan metode tradisional yang membutuhkan ekstraksi fitur secara manual, CNN dapat mempelajari pola langsung dari gambar yang diberikan. CNN banyak digunakan dalam berbagai bidang seperti pengenalan wajah, analisis gambar medis, dan pemrosesan bahasa alami (NLP), karena dirancang khusus untuk memahami bentuk-bentuk dalam gambar dua dimensi. Keunggulan utama CNN adalah kemampuannya untuk secara otomatis mengidentifikasi fitur penting dari gambar tanpa perlu intervensi manusia, sehingga lebih efisien dibandingkan jaringan saraf biasa.



Gambar 2.1. Komponen Model *Convolutional Neural Network*

Adapun lapisan-lapisan dalam sebuah model CNN, terdiri dari lapisan *input*, lapisan konvolusi, lapisan *pooling*, fungsi aktivasi, dan lapisan terhubung penuh (*fully connected*).

A. *Input Image*

Setiap gambar digital tersusun dari piksel yang memiliki nilai antara 0 hingga 255. Piksel ini membentuk sebuah matriks yang menentukan warna dan kecerahan gambar. Saat manusia melihat gambar, otak secara otomatis mengenali pola-pola yang ada. CNN bekerja dengan cara yang serupa, yaitu dengan mengenali pola sederhana terlebih dahulu (seperti garis dan bentuk dasar), lalu memahami objek yang lebih kompleks seperti wajah atau benda.

B. Lapisan Konvolusi (*Convolution Layer*)

Ini adalah lapisan utama dalam CNN yang bertugas mendeteksi fitur-fitur dalam gambar. Lapisan ini menggunakan filter (atau kernel) untuk mencari pola penting dalam gambar. Pada awalnya, nilai dalam filter ini ditentukan secara acak, tetapi selama pelatihan, nilai-nilai ini diperbaiki agar lebih akurat dalam mengenali pola tertentu. Hasil dari proses ini disebut *feature map*, yang merupakan representasi fitur penting dari gambar masukan.

C. Lapisan Pooling (*Pooling Layer*)

Lapisan *pooling* berfungsi untuk mengurangi ukuran data tanpa

menghilangkan informasi penting. Hal ini membantu mempercepat proses pelatihan model dan mengurangi kebutuhan penyimpanan. Salah satu metode *pooling* yang sering digunakan adalah *max pooling*, di mana gambar dibagi menjadi beberapa bagian kecil, dan hanya nilai tertinggi dari setiap bagian yang dipertahankan.

D. Fungsi Aktivasi (*Activation Function*)

Fungsi aktivasi berperan dalam menentukan apakah suatu neuron harus aktif atau tidak, sehingga model dapat memahami pola yang lebih kompleks. Fungsi ini juga membantu dalam proses pelatihan, terutama saat memperbaiki kesalahan prediksi menggunakan metode *backpropagation*.

E. *Fully connected Layer*

Ini adalah tahap akhir dalam CNN, di mana semua informasi dari lapisan sebelumnya dikumpulkan menjadi satu vektor. Vektor ini digunakan untuk menentukan hasil akhir dari model. Semua nilai dari lapisan sebelumnya dihitung ulang berdasarkan bobot dan bias, kemudian diproses menggunakan fungsi aktivasi untuk menentukan *output* yang paling sesuai. Jika hasilnya belum optimal, model akan memperbarui bobotnya hingga mencapai akurasi terbaik.

2.7. Penelitian Terkait

Terdapat beberapa penelitian yang relevan terkait penerapan teknik augmentasi, khususnya transformasi geometrik dan penerapan *Edge Detection* dalam mengolah dataset untuk memperkaya variasi data pada pengenalan gesture tangan (*Hand Gesture Recognition*) ataupun bidang lainnya. Dalam penelitian yang dilakukan oleh Awaluddin dkk. (2023) disorot bahwa pemanfaatan transformasi geometrik pada lima dataset HGR statis menggunakan tiga model pra-latih (ResNet50, MobileNetV2, dan InceptionV3), menunjukkan tidak semua transformasi geometrik memberikan manfaat yang signifikan. Secara khusus, image *shearing* dan horizontal *flipping* memiliki dampak paling besar dalam meningkatkan akurasi klasifikasi, dengan ResNet50 secara konsisten mengungguli dua model lainnya meskipun memerlukan lebih banyak daya komputasi. Selain itu, penelitian ini menemukan bahwa penggunaan transformasi yang terlalu kompleks dapat menurunkan performa model, terutama pada model yang lebih ringan, karena

variasi yang berlebihan dapat mengganggu kemampuan model dalam mengenali pola yang sebenarnya.

Penelitian yang dilakukan oleh de la Rosa dkk. (2022), mengkaji pengaruh augmentasi data berbasis transformasi geometrik terhadap kinerja klasifikasi cacat menggunakan model CNN ResNet50. Dengan menghasilkan hingga tujuh dataset berbeda, masing-masing memiliki jumlah gambar dua kali lipat dari dataset sebelumnya dan mengevaluasinya menggunakan metrik F1-score, penelitian menunjukkan bahwa prosedur augmentasi meningkatkan akurasi klasifikasi sebesar 3,74%. Peningkatan ini sangat signifikan terutama untuk dataset yang tidak seimbang, karena dengan menghasilkan gambar sintetis yang cukup untuk kelas minoritas, performa model meningkat secara signifikan. Selain itu, meskipun efek gangguan (*noise*) tidak diteliti karena dataset asli relatif bersih, penelitian menyimpulkan bahwa augmentasi data juga dapat membuat model lebih general dan mampu mengklasifikasikan gambar dengan kualitas lebih rendah dengan lebih baik.

Penelitian lain yang dilakukan oleh Mumuni dan Mumuni (2022), menegaskan bahwa *data augmentation* adalah strategi yang efektif dalam meningkatkan kinerja model pembelajaran mesin. *Geometric transformations* terbukti meningkatkan akurasi model dengan membuatnya lebih tahan terhadap perubahan tampilan objek. Dalam pengujian pada dataset CIFAR-10 menggunakan model ResNet-56, metode *random erasing* meningkatkan akurasi sebesar 1.5%, sementara teknik rotasi dan *shearing* masing-masing meningkatkan akurasi sebesar 1.14% dan 1.35%. Pada dataset ImageNet, kombinasi augmentasi seperti *CutMix* dan *SuperMix* meningkatkan top-1 accuracy hingga 3.6% dibandingkan *baseline*. Selain itu, kombinasi berbagai teknik augmentasi sering kali memberikan peningkatan akurasi yang lebih signifikan.

Selain itu, penelitian yang dilakukan oleh Kim dkk. (2023), mengevaluasi berbagai metode *thresholding* dalam kuantifikasi volume tumor PSMA-PET pada pasien kanker prostat lanjut. Fokus utama studi adalah membandingkan efektivitas metode *thresholding* dalam memprediksi *overall survival* (OS) pasien serta membandingkan akurasi dengan metode *baseline*. Dari penelitian ini *Thresholding* berbasis *Computer Vision* merupakan alat yang menjanjikan dalam kuantifikasi

volume tumor PSMA-PET. Meskipun penelitian ini berfokus pada *Thresholding* untuk segmentasi tumor, hasilnya menunjukkan bahwa metode seperti Otsu dan *Adaptive Thresholding* memiliki potensi besar dalam deteksi tepi (*Edge Detection*),

Terakhir, penelitian oleh Jardim dkk. (2023) mengulas berbagai pendekatan *thresholding* dalam segmentasi citra medis, yang merupakan teknik dasar namun efektif dalam pemrosesan citra medis. Penelitian ini menegaskan bahwa *thresholding* tetap menjadi metode yang sangat berguna dalam segmentasi citra medis, meskipun memiliki keterbatasan pada citra dengan iluminasi yang tidak merata dan objek yang kompleks. Metode Otsu, yang menentukan ambang optimal dengan memaksimalkan varians antar-kelas memiliki kelebihan cepat dan sederhana, tetapi rentan terhadap perubahan kontras dan pencahayaan yang tidak merata. *Adaptive Thresholding* lebih unggul dibandingkan metode *thresholding* global karena menangani citra dengan variasi pencahayaan atau objek kecil. Dari penelitian ini, *Otsu's Thresholding* menunjukkan hasil baik untuk citra dengan histogram bimodal, tetapi kurang efektif untuk citra dengan *noise* atau iluminasi tidak merata.

Tabel 2.1. Penelitian Terkait

No	Penulis	Judul	Tahun
1.	Baiti-Ahmad Awaluddin, Chun-Tang Chao, dan Juing-Shian Chiou	<i>Investigating Effective Geometric Transformation for Image Augmentation to Improve Static Hand Gestures with a Pre-Trained Convolutional Neural Network</i>	2023
2.	Francisco López de la Rosa, José L. Gómez-Sirvent, Roberto Sánchez-Reolid, Rafael Morales, dan Antonio Fernández-Caballero	<i>Geometric transformation-based data augmentation on defect classification of segmented images of semiconductor materials using a ResNet50 Convolutional Neural Network</i>	2022

No	Penulis	Judul	Tahun
3.	Alhassan Mumuni dan Fuseini Mumuni	<i>Data augmentation: A comprehensive survey of modern approaches</i>	2022
4.	Moon Kim, Robert Seifert, Jana Fragemann, David Kersting, Jacob Murray, Frederic Jonske, Kelsey L. Pomykala, Jan Egger, Wolfgang P. Fendler, Ken Herrmann, dan Jens Kleesiek	<i>Evaluation of Thresholding methods for the quantification of [68Ga] Ga-PSMA-11 PET molecular tumor volume and their effect on survival prediction in patients with advanced prostate cancer undergoing [177Lu] Lu-PSMA-617 radioligand therapy</i>	2023
5.	Sandra Jardim, João António, dan Carlos Mora	<i>Image Thresholding approaches for medical image segmentation - short literature review</i>	2023

BAB 3

ANALISIS DAN PERANCANGAN

3.1. Analisis Sistem

Analisis sistem merupakan langkah memecahkan dan menguraikan informasi menjadi bagian yang lebih kecil sehingga mudah dipahami. Proses analisis ini merupakan landasan awal sebelum melakukan perancangan dan pengembangan suatu sistem, agar sesuai dengan kebutuhan dan lebih terstruktur dalam mencapai tujuan akhir. Sistem ini mencakup berbagai tahapan, mulai dari *pre-processing* dataset menggunakan teknik Augmentasi Geometrik dan *Edge Detection*, hingga evaluasi hasil deteksi menggunakan model *Deep Learning* berbasis *Convolutional Neural Network* (CNN).

3.1.1. Analisis Masalah

Penelitian ini mengidentifikasi permasalahan dalam komunikasi antara penyandang disabilitas Tuli dan masyarakat non-Tuli, khususnya dalam konteks deteksi gerakan tangan pada Bahasa Isyarat Indonesia (BISINDO) dalam *video call real-time*. Untuk menganalisis permasalahan tersebut, penelitian ini menerapkan metode *5-Whys* dan menggambarkan Diagram Ishikawa.

Metode *5-Whys*, yaitu pendekatan tanya-jawab sederhana yang efektif dalam mengidentifikasi akar penyebab suatu masalah. Metode ini dilakukan dengan mengajukan pertanyaan "mengapa" secara berulang hingga ditemukan penyebab utama dari permasalahan yang dihadapi. Berikut adalah analisis menggunakan metode *5-Whys*:

1. Mengapa komunikasi antara penyandang disabilitas Tuli dan Masyarakat non-Tuli sering mengalami hambatan?
Karena terdapat variasi dalam gerakan tangan, kecepatan, dan intensitas ekspresi dalam Bahasa Isyarat Indonesia (BISINDO) yang sulit dikenali oleh sistem pendekripsi otomatis.
2. Mengapa variasi dalam gerakan tangan, kecepatan, dan ekspresi dapat menjadi masalah dalam sistem pendekripsi otomatis?

Karena model deteksi gerakan tangan berbasis *Deep Learning* membutuhkan data yang cukup beragam untuk dapat mengenali berbagai variasi tersebut dengan baik. Jika model tidak cukup dilatih dengan variasi gerakan, maka akurasi pendektsian akan menurun.

3. Mengapa model *Deep Learning* sulit mendekksi variasi gerakan tangan dengan baik dalam kondisi *video call real-time*?

Karena adanya faktor seperti perbedaan sudut pandang kamera, pencahayaan yang tidak konsisten, serta keterbatasan dataset dalam mencakup semua kemungkinan variasi gerakan tangan.

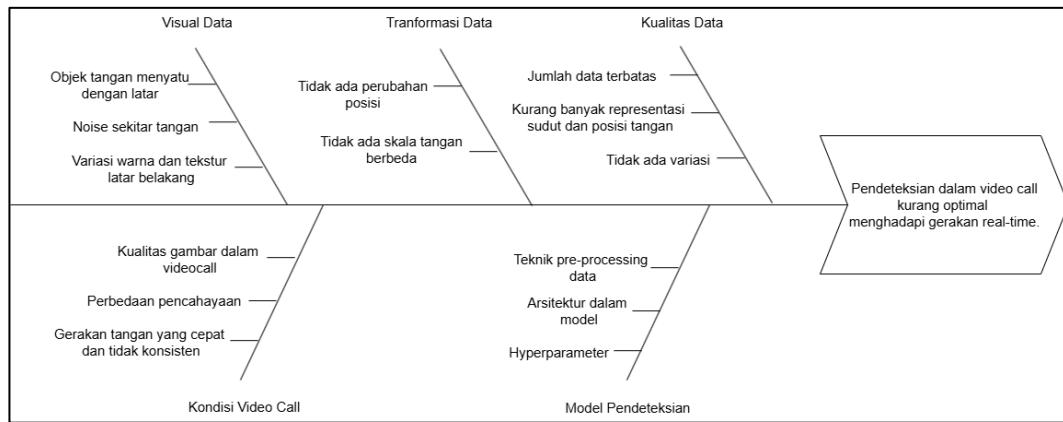
4. Mengapa perbedaan sudut pandang kamera dan pencahayaan yang tidak konsisten dapat menurunkan akurasi sistem?

Karena model pendektsian yang tidak cukup terlatih dengan berbagai kondisi pencahayaan dan sudut pandang akan kesulitan mengenali pola gerakan tangan secara akurat, yang dapat menyebabkan kesalahan interpretasi.

5. Mengapa teknik Augmentasi Geometrik dan *Edge Detection* dapat membantu meningkatkan akurasi sistem pendektsian?

Karena Augmentasi Geometrik memperbanyak variasi data pelatihan dengan mensimulasikan berbagai sudut pandang dan kondisi pencahayaan, sementara *Edge Detection* membantu menyoroti batasan kontur tangan sehingga sistem dapat lebih mudah mengenali bentuk dan pergerakan tangan dalam berbagai kondisi.

Diagram Ishikawa, atau dikenal sebagai *Fishbone Diagram*, adalah alat analisis yang digunakan untuk mengidentifikasi penyebab utama dari suatu permasalahan dalam sistem atau penelitian. Diagram ini membantu mengorganisir faktor-faktor yang berpengaruh terhadap suatu kendala, sehingga memudahkan dalam menemukan solusi yang tepat.



Gambar 3.1. Diagram Ishikawa

Diagram Ishikawa dalam penelitian ini menggambarkan faktor-faktor yang menyebabkan pendekstrian dalam *video call* ElCue kurang optimal dalam menghadapi variasi gerakan *real-time*. Faktor utama yang dianalisis meliputi visual data, transformasi data, kualitas data, kondisi *video call*, dan model pendekstrian. Masalah pada visual data seperti *noise*, warna, dan tekstur yang tidak konsisten dapat mengganggu proses ekstraksi fitur. Sementara itu, transformasi data yang tidak memvariasikan posisi dan skala tangan menyebabkan model sulit mengenali berbagai kemungkinan gerakan. Keterbatasan jumlah data dan kurangnya representasi sudut tangan juga menjadi kendala dalam kualitas data, sementara variasi dalam kualitas *video call* dan kamera pengguna mempengaruhi kondisi data latih. Selain itu, faktor teknis seperti *pre-processing*, arsitektur model, dan pemilihan *hyperparameter* juga berkontribusi terhadap ketidak sempurnaan dalam pendekstrian bahasa isyarat secara *real-time*.

3.1.2. Analisis Data

Dalam penelitian ini, dataset yang digunakan terdiri dari Bahasa Isyarat Indonesia (BISINDO) yang mencakup sembilan isyarat tangan. Dataset isyarat ini dikumpulkan penulis dengan didampingi langsung oleh Gerakan untuk Kesejahteraan Tunarungu Indonesia (GERKATIN) Provinsi Sumatera Utara. Dataset ini akan digunakan sebagai data awal sebelum melakukan *pre-processing* data. Berikut adalah analisis rinci terkait kriteria data, pengambilan data, dan jumlah data.

A. Kriteria Data

Dataset mencakup sembilan isyarat BISINDO, yaitu: “A”, “Asal”, “B”, “C”, “D”, “E”, “Halo”, “I”, “Terima Kasih”. Untuk setiap isyarat, dataset mencakup gambar (image) dengan gerakan tangan yang dilakukan oleh 5 individu, termasuk penulis. Dataset harus melibatkan variasi pencahayaan, sudut pandang, serta latar belakang yang berbeda untuk memastikan model yang dilatih dapat mengenali gerakan tangan dalam berbagai kondisi.

B. Pengambilan Data

Proses pengambilan data dilakukan menggunakan program pengumpulan data berbasis Python dengan pustaka OpenCV. Setelah data terkumpul, data diunggah ke dalam platform Kaggle untuk memudahkan akses dan penyimpanan. Dataset yang telah diunggah ke dalam platform Kaggle akan digunakan dalam program load data ke dalam sistem. Dalam proses ini, dataset yang terstruktur di Kaggle akan dimuat menggunakan API Kaggle ke dalam program.

C. Jumlah Data

Dataset terdiri dari 9488 gambar yang mencakup sembilan isyarat BISINDO. Jumlah data gambar dalam setiap isyarat yaitu: “A” sebanyak 1050 gambar, “Asal” sebanyak 1050 gambar, “B” sebanyak 1060 gambar, “C” sebanyak 1070 gambar, “D” sebanyak 1050 gambar, “E” sebanyak 1050 gambar, “Halo” sebanyak 1050 gambar, “I” sebanyak 1058 gambar, “Terima Kasih” sebanyak 1050 gambar.

3.1.3. Analisis Kebutuhan

Dalam penelitian ini, sistem yang dikembangkan bertujuan untuk meningkatkan akurasi model pengenalan gerakan BISINDO berbasis CNN melalui proses *pre-processing* dengan teknik Augmentasi Geometrik dan *Edge Detection*. Bagian ini memberikan gambaran yang jelas mengenai perangkat keras dan perangkat lunak yang dibutuhkan untuk penelitian ini.

A. Kebutuhan Perangkat Lunak

Untuk mendukung pengembangan sistem deteksi dan pengenalan gerakan BISINDO berbasis CNN dengan teknik *pre-processing* Augmentasi Geometrik dan *Edge Detection*, perangkat lunak yang digunakan terdapat

pada tabel 3.1. berikut.

Tabel 3.1. Kebutuhan Perangkat Lunak

Perangkat Lunak	Keterangan
Google Colab	v1.0.0, platform python berbasis <i>cloud</i>
Kaggle	Mengunduh dataset relevan
TensorFlow / Keras	v2.18.0
OpenCV	v4.11.0
Matplotlib	v3.10.0
Skicit-learn	v1.6.1
Numpy	v1.26.4
ImageDataGenerator	v2.18.0, keras.src.legacy.preprocessing.image

B. Kebutuhan Perangkat Keras

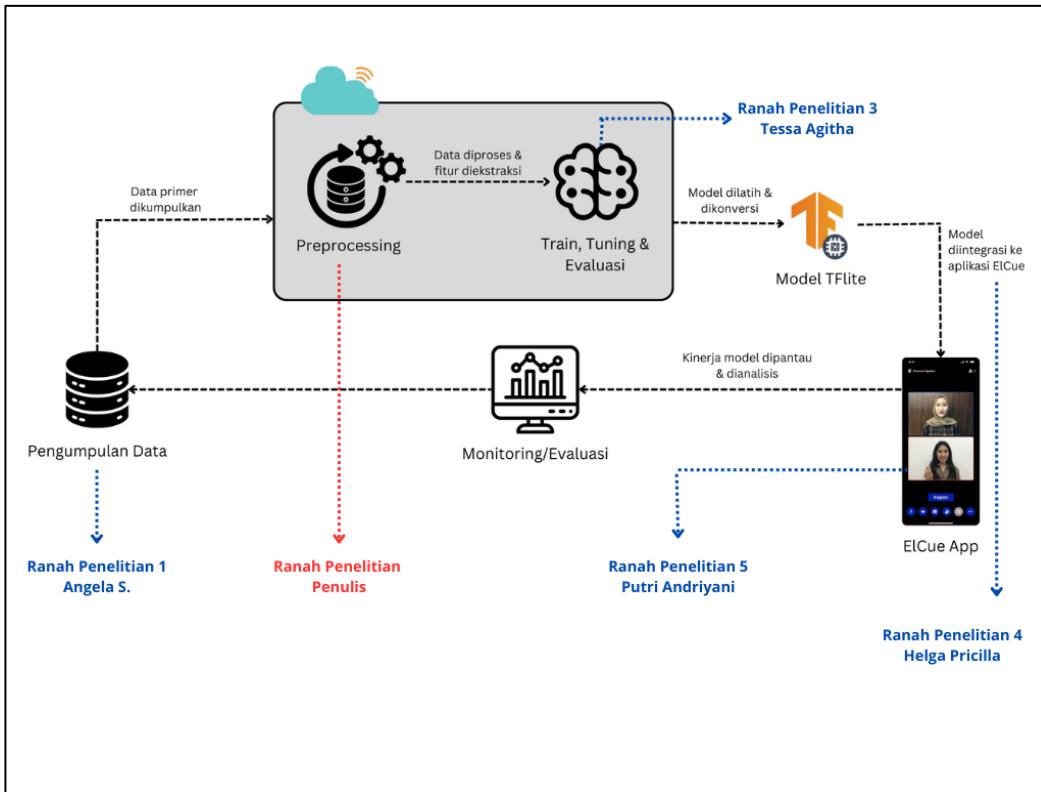
Berikut tabel 3.2. mendeskripsikan kebutuhan perangkat keras yang mendukung pengembangan sistem ini.

Tabel 3.2. Kebutuhan Perangkat Keras

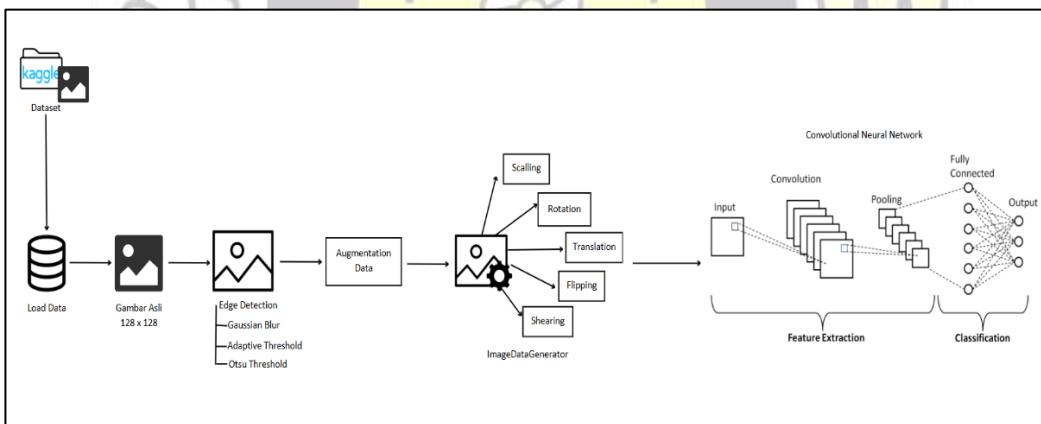
Perangkat Keras	Keterangan
CPU	AMD Ryzen 3 5300U
GPU	AMD Radeon(TM) Grapich
RAM	8 GB
Storage	SSD 512 GB

3.1.4. Arsitektur Umum Sistem

Dalam aplikasi ElCue, alur pengolahan data hingga implementasi model, terdiri dari empat tahap utama, yaitu: *Process Data, Develop Model, Deploy*, dan Monitor. Tahap *Process Data* mencakup pengumpulan dan pemrosesan data, yang menjadi fondasi awal dalam pengembangan model. Ranah penelitian penulis berfokus pada tahap *pre-processing* Data, yang bertujuan untuk meningkatkan kualitas *input* data sebelum model dilatih dan di-tuning agar mampu mengenali bahasa isyarat Bisindo secara *real-time* dalam aplikasi *video call* ElCue.



Gambar 3.1. Arsitektur Umum Aplikasi *Video call ElCue*



Gambar 3.2. Arsitektur Umum Sistem *Pre-processing*

Pada Gambar 3.2, arsitektur umum sistem menunjukkan alur proses dari awal, dimulai dari pengolahan dataset yang diperoleh dari Kaggle, yang kemudian melewati tahap *pre-processing* data untuk meningkatkan kualitas *input* sebelum digunakan dalam model. Pada tahap ini, digunakan teknik *Edge Detection* dengan metode *Gaussian blur*, *Adaptive Threshold*, dan *Otsu's Thresholding* untuk menonjolkan kontur tangan serta menghilangkan latar belakang yang tidak relevan. Setelah itu, data diproses lebih lanjut dengan *data augmentation* menggunakan

ImageDataGenerator, di mana gambar mengalami transformasi seperti *scaling*, *rotation*, *translation*, *flipping*, dan *shearing*. Augmentasi ini bertujuan untuk memperkaya variasi data latih, sehingga model dapat lebih *robust* terhadap perubahan posisi, sudut, dan ukuran tangan dalam bahasa isyarat.

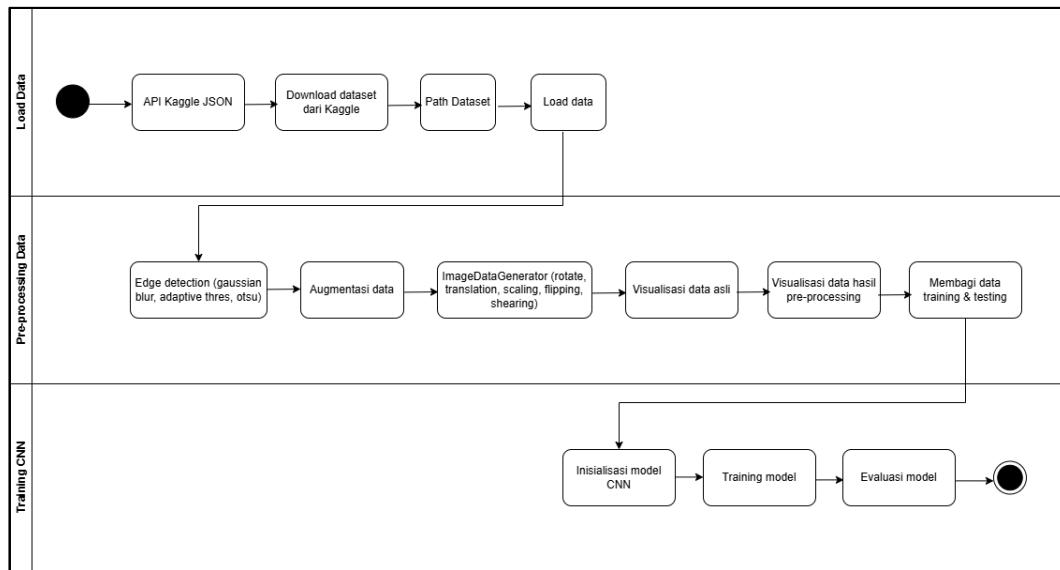
Setelah tahap *pre-processing*, gambar yang telah diolah dimasukkan ke dalam CNN untuk ekstraksi fitur dan klasifikasi. CNN pertama-tama melakukan *Feature Extraction* melalui *Convolution Layer* untuk mengenali pola dan bentuk isyarat tangan, diikuti oleh *Pooling Layer* yang mengurangi dimensi data sambil mempertahankan fitur penting. Hasil ekstraksi fitur kemudian diteruskan ke tahap *Classification*, di mana *Fully connected Layer* menghubungkan fitur yang diperoleh dengan *Output Layer*, yang bertugas mengklasifikasikan isyarat ke dalam kategori yang sesuai.

3.2. Pemodelan Sistem

Pemodelan sistem dalam penelitian ini bertujuan untuk menggambarkan alur kerja dan proses yang terjadi dalam sistem. Pemodelan ini mencakup beberapa diagram yang digunakan untuk mendukung analisis dan desain sistem, yaitu Diagram Ishikawa, *Use Case Diagram*, *Activity Diagram*, *Sequence Diagram* dan *Flowchart*.

3.2.1. *Activity Diagram*

Activity Diagram merupakan representasi grafis, dimana menggambarkan alur kerja sistem secara terstruktur, mulai dari proses awal hingga akhir. Diagram ini menunjukkan bagaimana suatu sistem mengeksekusi tugas-tugas dalam urutan tertentu, serta bagaimana berbagai komponen berinteraksi dalam sistem tersebut. Dalam konteks penelitian ini, *Activity Diagram* digunakan untuk memvisualisasikan tahapan utama dalam pemrosesan data dengan teknik augmentasi geometrik dan *edge detection* serta pelatihan model *Convolutional Neural Network* (CNN) untuk mendeteksi bahasa isyarat.



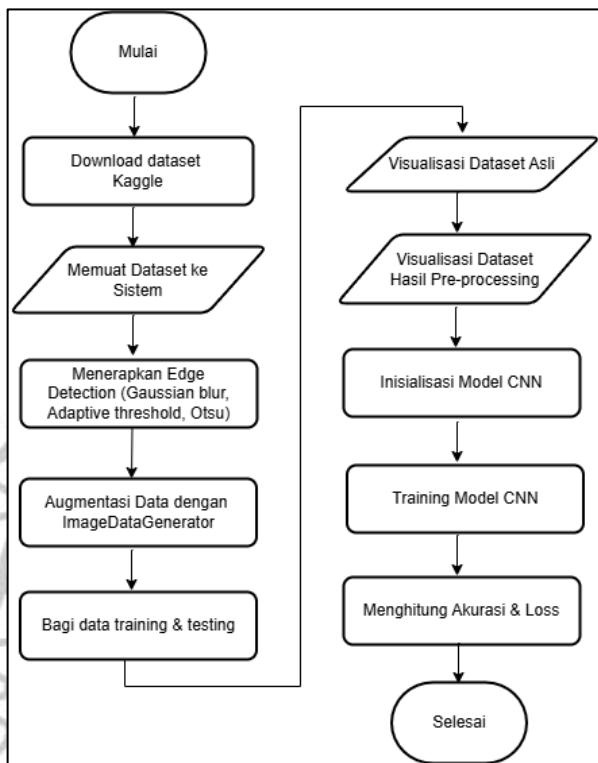
Gambar 3.3. *Activity Diagram*

Pada Gambar 3.3., *Activity Diagram* penelitian ini terdapat tiga swimlane utama, yaitu: *Load Data*, *Pre-processing Data*, dan *Training CNN*. Proses dimulai dengan mengunduh dataset dari Kaggle menggunakan API, kemudian memuat dataset ke dalam sistem. Setelah itu, dilakukan *pre-processing* data yang mencakup *Edge Detection* (*Gaussian blur*, *Adaptive Threshold*, *Otsu*) untuk menyoroti fitur tangan dalam gambar, diikuti oleh augmentasi data menggunakan *ImageDataGenerator* dengan transformasi seperti rotasi, translasi, *scaling*, *flipping*, dan *shearing*. Tahap selanjutnya adalah visualisasi dataset asli dan hasil *pre-processing*, sebelum membagi data menjadi *training* dan *testing*. Setelah data siap, model CNN diinisialisasi dan dilatih menggunakan dataset yang telah diproses. Terakhir, dilakukan evaluasi model untuk menilai performa dalam mengenali bahasa isyarat, sebelum sistem menyelesaikan eksekusinya. Diagram ini secara rinci menggambarkan aliran kerja penelitian dari awal hingga akhir, memastikan setiap langkah dilakukan secara sistematis untuk meningkatkan akurasi deteksi bahasa isyarat.

3.2.2. *Flowchart*

Flowchart merupakan representasi grafis yang menggambarkan alur proses dalam suatu sistem secara sistematis dan terstruktur. Dalam penelitian ini, *Flowchart* digunakan untuk mengilustrasikan tahapan utama dalam pemrosesan dataset, *pre-processing*, augmentasi, hingga pelatihan model CNN. Diagram ini memastikan

bahwa setiap langkah dalam penelitian dilakukan secara berurutan dan sistematis, sehingga model yang dihasilkan dapat bekerja secara optimal dalam mendeteksi isyarat BISINDO.



Gambar 3.4. Flowchart Penelitian

Gambar merupakan *flowchart* penelitian yang terdiri dari beberapa tahapan utama yang terbagi dalam pengolahan dataset, *pre-processing*, dan pelatihan model CNN. Proses dimulai dengan mengunduh dataset dari Kaggle, kemudian memuat dataset ke dalam sistem. Selanjutnya, dilakukan *Edge Detection (Gaussian blur, Adaptive Threshold, Otsu)* untuk menyoroti fitur penting dalam gambar. Setelah itu, dataset diproses lebih lanjut dengan *Augmentasi* menggunakan *ImageDataGenerator*, yang mencakup rotasi, translasi, *scaling*, *shearing*, dan *flipping*. Setelah proses *pre-processing* selesai, dataset dibagi menjadi data *training* dan *testing*, lalu model CNN diinisialisasi, dilatih, dan dievaluasi untuk memperoleh akurasi dan *loss* model. *Flowchart* ini juga mencakup tahapan visualisasi dataset asli dan hasil *pre-processing*, yang bertujuan untuk memastikan bahwa transformasi yang diterapkan sudah sesuai sebelum masuk ke tahap pelatihan model CNN.

BAB 4

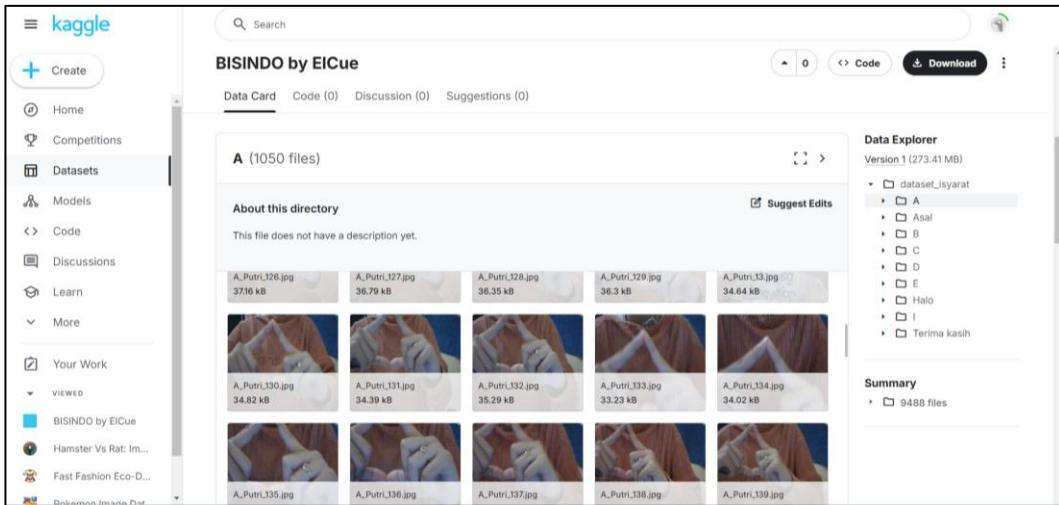
IMPLEMENTASI DAN PENGUJIAN

4.1. Implementasi Sistem

Implementasi sistem dalam penelitian ini mencakup penerapan langkah-langkah yang telah dirancang pada bab sebelumnya menerapkan *pre-processing* (*Edge Detection & Augmentasi*), serta melatih dan mengevaluasi dengan model CNN. Sistem ini dibangun dengan tujuan untuk meningkatkan kualitas data dan akurasi dalam pendekripsi isyarat BISINDO, khususnya dalam kondisi *video call real-time*. Proses implementasi dimulai dari *pre-processing* data, penerapan kedua teknik *pre-processing*, hingga pelatihan dan evaluasi model CNN. Bagian ini akan menjelaskan secara rinci bagaimana sistem diimplementasikan berdasarkan tahapan yang telah dirancang dalam penelitian ini.

4.1.1. *Pre-processing* Data

Pada tahap ini, sistem mulai dengan mengunduh dataset dari Kaggle menggunakan API Kaggle yang dikonfigurasi dalam lingkungan Google Colab. Setelah dataset berhasil diunduh, file diekstrak dan dimuat ke dalam sistem untuk diproses lebih lanjut. Dataset ini berisi gambar bahasa isyarat yang nantinya akan diklasifikasikan menggunakan model CNN. Dataset kemudian akan dikonversi ke dalam grayscale agar model lebih fokus pada fitur bentuk tanpa dipengaruhi oleh warna. Setiap gambar yang telah dimuat juga akan dinormalisasi dengan membagi nilai piksel dengan 255, sehingga nilainya berada dalam rentang 0 hingga 1, yang bertujuan untuk meningkatkan stabilitas *training* dan mempercepat proses konvergensi model. Berikut dokumentasi dataset yang di unduh melalui platform Kaggle:



Gambar 4.1. Dataset BISINDO dari Platform Kaggle

Dalam sistem ini, *pre-processing* dataset dilakukan menggunakan *Edge Detection* dan Augmentasi Geometrik untuk meningkatkan kualitas dan variasi data sebelum digunakan dalam pelatihan model. *Edge Detection* diterapkan terlebih dahulu untuk mempertajam fitur tangan dengan mengurangi latar belakang yang tidak diperlukan, sehingga model lebih fokus pada bentuk dan kontur tangan. Setelah itu, dilakukan Augmentasi Geometrik untuk meningkatkan variasi dataset guna meningkatkan kemampuan generalisasi model. Proses augmentasi ini nantinya diikuti dengan pembagian dataset menjadi data *training* (80%) dan data *testing* (20%), yang memungkinkan model untuk belajar dari berbagai variasi data dan diuji dengan data yang belum pernah dilihat sebelumnya.

Berikut potongan kode untuk mengunduh dataset dan memuat data sebelum dilakukan *pre-processing*:

1. Konfigurasi Kaggle API

Kode ini digunakan untuk mengunduh dataset dari Kaggle ke dalam Google Colab.

```
!pip install -q Kaggle
from google.colab import files
files.upload()
! mkdir ~/.kaggle
! cp kaggle.json ~/.kaggle/
! chmod 600 ~/.kaggle/kaggle.json
! kaggle datasets list
```

Gambar 4.2. Program Konfigurasi Kaggle

2. Unduh dan Ekstrak Dataset

Dataset bahasa isyarat Bisindo-by-ElCue diunduh dan diekstrak.

```
# download dataset
!kaggle datasets download -d tessaagitha/bisindo-by-elcue
# unzip dataset
!mkdir bisindo-by-elcue
!unzip bisindo-by-elcue.zip -d bisindo-by-elcue
!ls bisindo-by-elcue
```

Gambar 4.3. Program Unduh Dataset

3. Impor Pustaka yang Dibutuhkan

Memuat pustaka Python yang dibutuhkan untuk *pre-processing*, augmentasi, dan pelatihan model.

```
import os
import shutil
import cv2 # Import the OpenCV library
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
import visualkeras
import tensorflow.keras as keras
import sklearn.metrics as metrics
import seaborn as sns
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import img_to_array
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.optimizers import Adam
```

Gambar 4.4. Program Impor Library

Kode di atas mengimpor berbagai pustaka yang digunakan dalam pemrosesan data, augmentasi, serta pelatihan model *Convolutional Neural Network* (CNN). OpenCV (cv2) digunakan untuk membaca dan memproses gambar, termasuk penerapan *Edge Detection*. NumPy menangani operasi numerik dan manipulasi *array*. Matplotlib & Seaborn digunakan untuk visualisasi data, termasuk *plotting* hasil *pre-processing* dan kinerja model. Scikit-learn (sklearn) menyediakan fungsi *train_test_split* untuk membagi dataset dan *metrics* untuk evaluasi model. TensorFlow/Keras digunakan untuk membangun dan melatih model CNN, dengan Sequential API untuk

mendefinisikan arsitektur, *Layers* untuk menambahkan lapisan, dan Adam optimizer untuk mempercepat konvergensi. *ImageDataGenerator* membantu dalam augmentasi data guna meningkatkan keberagaman dataset.

4. Path Dataset

Selanjutnya menentukan lokasi folder dataset yang telah diekstrak. Gunakan folder dataset ini untuk mengambil data dalam langkah selanjutnya.

5. Memuat dan Memproses Dataset

Membaca gambar, menerapkan *Edge Detection*, mengubah ukuran gambar, serta menormalisasi data.

```
def load_images():
    images = [] labels = [] index = -1
    folders = sorted(os.listdir(DATASET_PATH))
    for folder in folders:
        index += 1
        print(f"Loading images from folder: {folder}")
        folder_path = os.path.join(DATASET_PATH, folder)
        for image_name in os.listdir(folder_path):
            image_path = os.path.join(folder_path, image_name)
            img = cv2.imread(image_path, 0)
            if img is not None:
                img = edge_detection(img)
                img = cv2.resize(img, (64, 64))
                img = img_to_array(img)
                images.append(img)
                labels.append(index)
    images = np.array(images, dtype="float32") / 255.0
    labels = to_categorical(labels)
    x_train, x_test, y_train, y_test = train_test_split(images, labels,
    test_size=0.2, random_state=42)
    return x_train, x_test, y_train, y_test
```

Gambar 4.5. Program Memuat Dataset

4.1.2. Penerapan *Edge Detection*

Penerapan *Edge Detection* pada gambar bahasa isyarat bertujuan menyoroti kontur tangan serta menghilangkan gangguan latar belakang. Proses ini dimulai dengan menerapkan *Gaussian blur* menggunakan kernel (5x5) untuk mengurangi *noise* dalam gambar sebelum proses *Thresholding* dilakukan. Setelah itu, digunakan *Adaptive Thresholding* dengan metode *Gaussian Adaptive* yang secara otomatis menyesuaikan nilai ambang berdasarkan intensitas lokal pada gambar, membantu

mempertajam perbedaan antara objek tangan dan latar belakang. Langkah terakhir adalah menerapkan Otsu's *Thresholding*, yang secara otomatis menentukan nilai ambang optimal berdasarkan histogram piksel dalam gambar. Berikut potongan kode untuk menerapkan deteksi tepi (*Edge Detection*):

```
# Fungsi Edge Detection
def edge_detection(image):
    minValue = 70
    blur = cv2.GaussianBlur(image, (5, 5), 2)
    th3 = cv2.adaptiveThreshold(blur, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                cv2.THRESH_BINARY_INV, 11, 2)
    ret, res = cv2.Threshold(th3, minValue, 255, cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)
    return res
```

Gambar 4.6. Program Penerapan *Edge Detection*

1. Gaussian blur

Penerapan *Gaussian blur* dari OpenCV dengan Persamaan (10) menghasilkan konvolusi antara gambar asli dengan fungsi Gaussian, yang berfungsi untuk menghaluskan gambar dan mengurangi *noise*. Kernel Gaussian (5x5) dihitung menggunakan Persamaan (10) dengan $\sigma = 2$, menghasilkan matriks bobot Gaussian yang menentukan seberapa besar kontribusi setiap piksel dalam proses blur. Setiap piksel dalam gambar dikalikan dengan bobot dalam kernel dan dijumlahkan, menghasilkan nilai rata-rata berbobot yang menghaluskan perubahan intensitas pada gambar.

2. Adaptive Thresholding

Fungsi ini membagi gambar menjadi area kecil (blok berukuran 11×11) dan menetapkan nilai ambang secara lokal. Selanjutnya, metode *Gaussian Adaptive* menentukan ambang batas setiap blok berdasarkan rata-rata bobot Gaussian di sekitarnya, sehingga mampu menangani perubahan intensitas cahaya di seluruh gambar. Mode *THRESH_BINARY_INV* akan membalik hasil, sehingga piksel tangan menjadi putih (255) dan latar belakang menjadi hitam (0).

3. Otsu Thresholding

Fungsi ini menghitung distribusi histogram intensitas piksel, lalu mencari ambang batas yang meminimalkan varians intra-kelas (selisih antara piksel *foreground* dan *background*). Nilai $\text{minValue} = 70$ adalah batas bawah yang

digunakan sebagai ambang sementara sebelum *Otsu's Thresholding* menentukan *threshold* optimal secara otomatis.

Setelah fungsi `edge_detection` dibuat, dataset bahasa isyarat dimuat ke dalam sistem, di mana setiap gambar akan diproses dengan fungsi tersebut.

```
# Memuat dataset
x_train, x_test, y_train, y_test = load_images()
```

Gambar 4.7. Program *Load Data*

Selanjutnya, data diacak untuk memastikan bahwa urutan gambar tidak menyebabkan bias selama *training*.

```
from sklearn.utils import shuffle
# Mengacak data
x_train, y_train = shuffle(x_train, y_train, random_state=17)
x_test, y_test = shuffle(x_test, y_test, random_state=17)
```

Gambar 4.8. Program Mengacak Data Latih dan Uji

4.1.3. Penerapan Transformasi Geometrik

Penerapan augmentasi data ini bertujuan untuk meningkatkan variasi dataset dengan melakukan berbagai transformasi geometrik, sehingga model lebih *robust* dan mampu mengenali pola dalam berbagai kondisi. Augmentasi ini dilakukan dengan menerapkan `ImageDataGenerator`. `ImageDataGenerator` adalah kelas dalam TensorFlow/Keras yang digunakan untuk melakukan augmentasi data secara *real-time* selama proses pelatihan model. Fitur utama dari `ImageDataGenerator` adalah kemampuannya untuk menghasilkan variasi baru dari gambar asli dengan menerapkan berbagai transformasi seperti rotasi, translasi, *zoom*, *flipping*, *shearing*, dan normalisasi. Augmentasi ini membantu memperkaya dataset tanpa perlu menambah jumlah data secara manual. Berikut ini potongan kode penerapan transformasi geometrik:

```
# Augmentasi Data
train_datagen = ImageDataGenerator(
    rotation_range=15,
    width_shift_range=0.1,
    height_shift_range=0.1,
    zoom_range=0.2,
    shear_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)
test_datagen = ImageDataGenerator()
train_generator = train_datagen.flow(x_train, y_train, batch_size=32, shuffle=True)
test_generator = test_datagen.flow(x_test, y_test, batch_size=32, shuffle=False)
```

Gambar 4.9. Program Penerapan Augmentasi

1. *Rotation* (rotation_range=15)

Rotasi gambar hingga ± 15 derajat dilakukan dengan memutar titik (x, y) di sekitar pusat rotasi (cx, cy) berdasarkan sudut θ . Dalam kode ini, gambar dapat diputar hingga $\pm 15^\circ$ secara acak selama proses augmentasi. Objek (tangan dalam bahasa isyarat) diputar searah atau berlawanan jarum jam.

2. *Translation* (width_shift_range=0.1, height_shift_range=0.1)

Translasi menggeser gambar horizontal dan vertikal berdasarkan nilai perubahan dx dan dy. width_shift_range=0.1 menggeser gambar ke kiri atau ke kanan hingga 10% dari lebar gambar. Sedangkan, height_shift_range=0.1 menggeser gambar ke atas atau ke bawah hingga 10% dari tinggi gambar. Hasilnya gambar akan tampak berpindah ke kiri, kanan, atas, atau bawah.

3. *Scaling* (zoom_range=0.2)

Scaling mengubah ukuran gambar dengan faktor skala s_x, s_y . zoom_range=0.2 berarti gambar bisa diperbesar atau diperkecil hingga 20% dari ukuran aslinya. Jika diperbesar, bagian tengah gambar tetap terlihat, tetapi bagian tepi mungkin keluar dari frame. Jika diperkecil, gambar tampak lebih kecil di tengah frame.

4. *Shearing* (shear_range=0.1)

Shearing adalah transformasi yang menciptakan efek miring (skewing) menggunakan sh_x, sh_y . shear_range=0.1 berarti gambar bisa dimiringkan

hingga 10% dari ukuran aslinya. Hasilnya objek akan tetap dalam ukuran yang sama tetapi tampak lebih condong atau miring.

5. *Flipping* (horizontal_flip=True)

Teknik ini akan membalik gambar secara horizontal, artinya gambar memiliki 50% kemungkinan untuk dibalik ke arah horizontal. Gambar akan dicerminkan seperti melihat ke kaca.

Selain itu, metode `fill_mode='nearest'` memastikan bahwa area kosong akibat transformasi diisi dengan interpolasi yang sesuai. Augmentasi ini hanya diterapkan pada data *training*, sedangkan data *testing* hanya dinormalisasi menggunakan `ImageDataGenerator()` tanpa transformasi tambahan, sehingga evaluasi model tetap objektif.

4.1.4. Model CNN untuk Pendekatan

Pada bagian ini, model *Convolutional Neural Network* (CNN) digunakan untuk melakukan klasifikasi bahasa isyarat berdasarkan gambar hasil *pre-processing*. Terdapat beberapa tahapan dalam implementasi model CNN dalam sistem ini, yaitu:

1. *Mapping* Label ke Karakter Bahasa Isyarat

Setiap kelas bahasa isyarat diberi label numerik, yang kemudian dipetakan ke karakter atau kata tertentu, seperti 0 untuk "A", 1 untuk "Asal", hingga 8 untuk "Terima Kasih". Dengan pemetaan ini, model dapat menerjemahkan hasil prediksinya ke dalam bentuk yang lebih mudah dipahami.

2. Arsitektur Model CNN

Model CNN yang digunakan dalam penelitian ini dirancang dengan arsitektur berlapis untuk mengekstrak fitur penting dari gambar bahasa isyarat. Model ini dibangun menggunakan *Sequential API* dari Keras, yang memungkinkan susunan lapisan secara berurutan.

```

model = Sequential([
    Layers.Conv2D(32, (3, 3), activation='relu', padding="same",
    input_shape=(64, 64, 1)),
    Layers.MaxPool2D((2, 2)),
    Layers.Conv2D(64, (3, 3), padding="same", activation='relu'),
    Layers.MaxPool2D((2, 2)),
    Layers.Conv2D(64, (3, 3), padding="same", activation='relu'),
    Layers.MaxPool2D((2, 2)),
    Layers.Conv2D(64, (3, 3), padding="same", activation='relu'),
    Layers.MaxPool2D((2, 2)),
    Layers.Flatten(),
    Layers.Dense(128, activation='relu'),
    Layers.Dense(9, activation='softmax')])

```

Gambar 4.10. Program Arsitektur Model CNN

Arsitektur CNN yang digunakan dalam penelitian ini terdiri dari beberapa lapisan konvolusi dan lapisan *pooling* untuk mengekstrak fitur penting dari gambar. Lapisan pertama adalah *Convolutional Layer* dengan 32 filter ukuran 3×3 , yang berfungsi untuk mendekripsi pola dasar seperti tepi dan sudut dalam gambar. Aktivasi ReLU (*Rectified Linear Unit*) digunakan untuk meningkatkan non-linearitas model, sehingga dapat menangkap pola kompleks dalam data. Setelah konvolusi, diterapkan *MaxPooling* 2×2 , yang berfungsi untuk mengurangi dimensi gambar sambil mempertahankan fitur penting. Proses ini diulang dalam beberapa lapisan, dengan jumlah filter yang meningkat menjadi 64, memungkinkan model menangkap fitur yang lebih kompleks kedalam jaringan.

Setelah melalui beberapa lapisan konvolusi dan *pooling*, gambar yang telah diproses kemudian diratakan (*Flatten Layer*) menjadi vektor satu dimensi, yang kemudian diteruskan ke lapisan Dense (*Fully connected Layer*). Lapisan ini terdiri dari 128 neuron yang bertanggung jawab untuk menggabungkan semua fitur yang telah diekstrak sebelumnya. Akhirnya, model memiliki lapisan *output* dengan 9 neuron, sesuai dengan jumlah kelas bahasa isyarat yang akan diklasifikasikan. Aktivasi *Softmax* digunakan pada lapisan *output* untuk mengubah hasil prediksi menjadi probabilitas untuk setiap kelas, sehingga model dapat menentukan gestur yang paling mungkin dari *input* yang diberikan.

3. Kompilasi Model

Setelah model dirancang, tahap selanjutnya adalah kompilasi model, yaitu

menyiapkan model sebelum pelatihan dengan menentukan jumlah kelas, jumlah iterasi pelatihan, metode optimasi, fungsi *loss*, dan metrik evaluasi.

```
classes = 9
epochs = 50
adam = Adam(learning_rate=0.0001)
model.compile(optimizer=adam, loss='categorical_crossentropy',
metrics=['accuracy'])
```

Gambar 4.11. Program Kompilasi Model CNN

Model memiliki 9 kelas *output*, yang sesuai dengan jumlah kategori bahasa isyarat yang akan diklasifikasikan. Model akan dilatih selama 50 *epoch*, yang berarti data akan melewati jaringan sebanyak 50 kali untuk memperbarui bobot dan meningkatkan akurasi prediksi. Selanjutnya, menggunakan Adam untuk memperbarui bobot model secara efisien selama *training* dan *categorical_crossentropy*, karena model menangani klasifikasi multi-kelas (9 kelas), *metrics*=['accuracy'] berarti model akan mengukur akurasi selama *training* dan validasi.

4. Proses *Training* Model

Proses *training* model dilakukan menggunakan fungsi *model.fit()*, yang berfungsi untuk melatih model CNN dengan dataset yang telah dilakukan *pre-processing* sebelumnya. Data *training* diambil dari *train_generator*, yang secara dinamis menghasilkan *batch* gambar hasil augmentasi selama proses pelatihan.

```
history = model.fit(
    train_generator,
    epochs=50,
    validation_data=test_generator,
    verbose=2, shuffle=True)
```

Gambar 4.12. Program *Training* Model CNN

Hasil *training* model CNN dengan menggunakan dataset hasil *pre-processing*, yaitu:

Epoch 1/50
238/238 - 70s - 294ms/step - accuracy: 0.5945 - loss: 1.1406 - val_accuracy: 0.5221 - val_loss: 1.2897
Epoch 2/50
238/238 - 67s - 280ms/step - accuracy: 0.6157 - loss: 1.0817 - val_accuracy: 0.5859 - val_loss: 1.1049
Epoch 3/50
238/238 - 68s - 285ms/step - accuracy: 0.6383 - loss: 1.0302 - val_accuracy: 0.6270 - val_loss: 0.9954
Epoch 4/50
238/238 - 72s - 301ms/step - accuracy: 0.6527 - loss: 0.9811 - val_accuracy: 0.6296 - val_loss: 1.0380
Epoch 5/50
238/238 - 78s - 326ms/step - accuracy: 0.6694 - loss: 0.9323 - val_accuracy: 0.6928 - val_loss: 0.9193
Epoch 6/50
238/238 - 67s - 283ms/step - accuracy: 0.6763 - loss: 0.8987 - val_accuracy: 0.5843 - val_loss: 1.0882
Epoch 7/50
238/238 - 64s - 313ms/step - accuracy: 0.6918 - loss: 0.8752 - val_accuracy: 0.6965 - val_loss: 0.8159
Epoch 8/50
238/238 - 111s - 468ms/step - accuracy: 0.7119 - loss: 0.8087 - val_accuracy: 0.6681 - val_loss: 0.9653
Epoch 9/50
238/238 - 67s - 283ms/step - accuracy: 0.7267 - loss: 0.7759 - val_accuracy: 0.6117 - val_loss: 1.0800
Epoch 10/50
238/238 - 66s - 278ms/step - accuracy: 0.7398 - loss: 0.7505 - val_accuracy: 0.6670 - val_loss: 0.9514
Epoch 11/50
238/238 - 68s - 284ms/step - accuracy: 0.7473 - loss: 0.7277 - val_accuracy: 0.6918 - val_loss: 0.8496
Epoch 12/50
238/238 - 66s - 278ms/step - accuracy: 0.7561 - loss: 0.6944 - val_accuracy: 0.7281 - val_loss: 0.7702
Epoch 13/50
238/238 - 68s - 287ms/step - accuracy: 0.7672 - loss: 0.6699 - val_accuracy: 0.7376 - val_loss: 0.7298
Epoch 14/50
238/238 - 66s - 277ms/step - accuracy: 0.7736 - loss: 0.6489 - val_accuracy: 0.6865 - val_loss: 0.8576
Epoch 15/50
238/238 - 67s - 280ms/step - accuracy: 0.7929 - loss: 0.6195 - val_accuracy: 0.7339 - val_loss: 0.7565
Epoch 16/50
238/238 - 68s - 287ms/step - accuracy: 0.7879 - loss: 0.6660 - val_accuracy: 0.6976 - val_loss: 0.9179
Epoch 17/50
238/238 - 67s - 281ms/step - accuracy: 0.7999 - loss: 0.5849 - val_accuracy: 0.7360 - val_loss: 0.7571
Epoch 18/50
238/238 - 67s - 283ms/step - accuracy: 0.8091 - loss: 0.5623 - val_accuracy: 0.7366 - val_loss: 0.7446

Epoch 19/50
238/238 - 66s - 277ms/step - accuracy: 0.8609 - loss: 0.4031 - val_accuracy: 0.8172 - val_loss: 0.5142
Epoch 33/50
238/238 - 82s - 344ms/step - accuracy: 0.8694 - loss: 0.3823 - val_accuracy: 0.8293 - val_loss: 0.4655
Epoch 34/50
238/238 - 67s - 280ms/step - accuracy: 0.8748 - loss: 0.3774 - val_accuracy: 0.8298 - val_loss: 0.5013
Epoch 35/50
238/238 - 67s - 282ms/step - accuracy: 0.8762 - loss: 0.3682 - val_accuracy: 0.8277 - val_loss: 0.5191
Epoch 36/50
238/238 - 67s - 280ms/step - accuracy: 0.8817 - loss: 0.3544 - val_accuracy: 0.8209 - val_loss: 0.5555
Epoch 37/50
238/238 - 66s - 276ms/step - accuracy: 0.8785 - loss: 0.3602 - val_accuracy: 0.7687 - val_loss: 0.7878
Epoch 38/50
238/238 - 66s - 279ms/step - accuracy: 0.8814 - loss: 0.3417 - val_accuracy: 0.8246 - val_loss: 0.5239
Epoch 39/50
238/238 - 68s - 285ms/step - accuracy: 0.8867 - loss: 0.3334 - val_accuracy: 0.7866 - val_loss: 0.6533
Epoch 40/50
238/238 - 66s - 276ms/step - accuracy: 0.8896 - loss: 0.3249 - val_accuracy: 0.8346 - val_loss: 0.4719
Epoch 41/50
238/238 - 66s - 276ms/step - accuracy: 0.8974 - loss: 0.3115 - val_accuracy: 0.8483 - val_loss: 0.4833
Epoch 42/50
238/238 - 64s - 271ms/step - accuracy: 0.9011 - loss: 0.2995 - val_accuracy: 0.8124 - val_loss: 0.6134
Epoch 43/50
238/238 - 66s - 279ms/step - accuracy: 0.8943 - loss: 0.3158 - val_accuracy: 0.8188 - val_loss: 0.5472
Epoch 44/50
238/238 - 70s - 294ms/step - accuracy: 0.8995 - loss: 0.3001 - val_accuracy: 0.8404 - val_loss: 0.4533
Epoch 45/50
238/238 - 66s - 275ms/step - accuracy: 0.9058 - loss: 0.2824 - val_accuracy: 0.8625 - val_loss: 0.4086
Epoch 46/50
238/238 - 62s - 344ms/step - accuracy: 0.8982 - loss: 0.2867 - val_accuracy: 0.8493 - val_loss: 0.4666
Epoch 47/50
238/238 - 66s - 278ms/step - accuracy: 0.8987 - loss: 0.3031 - val_accuracy: 0.8282 - val_loss: 0.4818
Epoch 48/50
238/238 - 66s - 276ms/step - accuracy: 0.9050 - loss: 0.2841 - val_accuracy: 0.8435 - val_loss: 0.4176
Epoch 49/50
238/238 - 68s - 286ms/step - accuracy: 0.9041 - loss: 0.2826 - val_accuracy: 0.8256 - val_loss: 0.5076
Epoch 50/50
238/238 - 66s - 278ms/step - accuracy: 0.9095 - loss: 0.2690 - val_accuracy: 0.8377 - val_loss: 0.5487

Gambar 4.13. Proses Training Model CNN

Pada proses *training* ini, *train_generator* menghasilkan *batch* gambar yang telah mengalami augmentasi sesuai dengan pengaturan dalam *train_datagen*. Setiap *batch* terdiri dari 32 gambar yang telah diterapkan rotasi, translasi, *scaling*, *shearing*, dan *flipping* secara acak. Total data augmentasi menjadi:

Total data augmentasi = $\frac{\text{jumlah data training}}{\text{batch size}} \times \text{epoch}$

Karena dataset awal sebelum augmentasi berjumlah 9.488 gambar, maka x_train sebesar 80% = $0.8 \times 9.488 = 7.590$ gambar

Total batch per epoch menjadi = $\frac{7590}{32} = 238$

Sehingga total data setelah augmentasi:

Total data augmentasi = $238 \times 32 \times 50 = 380.800$ gambar

Selain itu, validation data = test generator digunakan untuk

mengevaluasi performa model pada data yang tidak digunakan dalam *training*, sehingga dapat mengukur sejauh mana model mampu melakukan generalisasi terhadap data baru.

4.2. Pengujian Sistem

Setelah model *Convolutional Neural Network* (CNN) dibangun dan dilatih, tahap selanjutnya adalah pengujian sistem untuk mengevaluasi kinerja *pre-processing* data dan performa model dalam mengenali bahasa isyarat. Pengujian ini bertujuan untuk memastikan bahwa sistem mampu mengolah gambar bahasa isyarat secara optimal sebelum digunakan untuk klasifikasi, serta untuk mengukur sejauh mana model dapat mengenali pola isyarat dengan akurasi tinggi. Pengujian sistem ini dibagi menjadi dua bagian utama. Pertama, dilakukan pengujian terhadap dataset hasil *pre-processing*, yang mencakup *Edge Detection* dan Augmentasi Data. Bagian kedua dari pengujian sistem adalah pengujian performa model CNN.

4.2.1. Pengujian Dataset Hasil *Pre-processing*

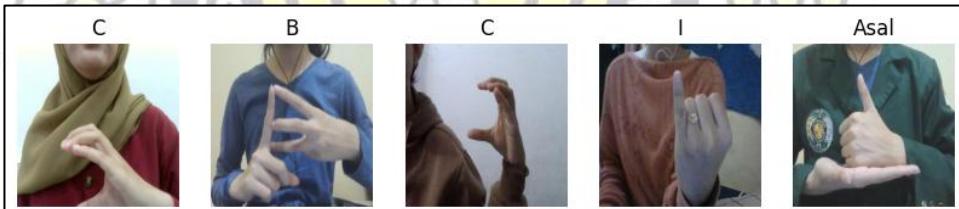
Pada bagian pengujian ini, dilakukan visualisasi terhadap data asli, data hasil *Edge Detection*, dan data hasil augmentasi. Tujuan dari pengujian ini adalah untuk memastikan bahwa setiap tahap *pre-processing* berjalan dengan baik dan tidak menghilangkan informasi penting dari gambar bahasa isyarat. Pengujian dilakukan dengan membandingkan gambar asli sebelum *pre-processing* dengan gambar setelah *Edge Detection* dan setelah augmentasi, guna mengevaluasi bagaimana transformasi ini memengaruhi kualitas data.

1. Visualisasi Dataset Asli

```
# Visualisasi Dataset Asli
import os
import random
import cv2
import matplotlib.pyplot as plt
def visualize_original_data(dataset_path, num_samples=5):
    classes = os.listdir(dataset_path)
    plt.figure(figsize=(10, num_samples * 2))
    for i in range(num_samples):
        class_name = random.choice(classes)
        class_path = os.path.join(dataset_path, class_name)
        img_name = random.choice(os.listdir(class_path))
        img_path = os.path.join(class_path, img_name)
        img = cv2.imread(img_path)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        plt.subplot(1, num_samples, i + 1)
        plt.imshow(img)
        plt.title(class_name)
        plt.axis('off')
    plt.show()
visualize_original_data(DATASET_PATH)
```

Gambar 4.14. Program Visualisasi Data Asli

Fungsi ini digunakan untuk menampilkan beberapa sampel gambar asli dari dataset sebelum dilakukan *pre-processing*, seperti *Edge Detection* dan augmentasi. Fungsi ini secara acak memilih 5 gambar dari berbagai kelas bahasa isyarat. *Output* dari fungsi ini, yaitu:



Gambar 4.15. Visualisasi Dataset Asli

2. Visualisasi Hasil *Edge Detection*

```
import random
unique_labels = np.unique(y_train.argmax(axis=1))
num_samples = 3
plt.figure(figsize=(15, 10))
for idx, label in enumerate(unique_labels):
    class_indices = np.where(y_train.argmax(axis=1) == label)[0]
    random_indices = random.sample(list(class_indices), min(num_samples,
    len(class_indices)))
    for j, img_idx in enumerate(random_indices):
        plt.subplot(len(unique_labels), num_samples, idx * num_samples + j + 1)
        plt.imshow(x_train[img_idx].squeeze(), cmap="gray")
        plt.axis("off")
        if j == 0:
            plt.title(f"Class {label}")
    plt.tight_layout()
plt.show()
```

Gambar 4.16. Program Visualisasi Hasil *Edge Detection*

Fungsi ini digunakan untuk menampilkan hasil *Edge Detection* dari dataset *pre-processing*, di mana algoritma *Gaussian blur*, *Adaptive Thresholding*, dan Otsu's *Thresholding* telah diterapkan untuk menyorot kontur tangan dalam skala grayscale. Kode ini memilih 3 gambar acak dari setiap kelas, lalu menampilkannya. *Output* dari fungsi ini, yaitu:



Gambar 4.17. Visualisasi Hasil *Edge Detection*

Hasil *Edge Detection* yang ditampilkan menunjukkan gambar bahasa isyarat dalam bentuk kontur hitam-putih setelah diterapkan *Gaussian Blur*, *Adaptive Thresholding*, dan *Otsu's Thresholding*. Dari gambar ini, dapat dianalisis beberapa aspek utama terkait efektivitas deteksi tepi pada dataset:

- Sebagian besar gambar berhasil menyoroti batas tangan dan bentuk jari dengan jelas.
- Beberapa gambar masih menunjukkan *noise* atau *detail* latar belakang,

meskipun dalam jumlah yang lebih sedikit dibandingkan gambar asli.

- Sebagian gambar terlihat lebih jelas dibandingkan yang lain, terutama yang memiliki kontras tinggi antara tangan dan latar belakang.
- Secara keseluruhan, bentuk isyarat tetap terjaga, menunjukkan bahwa proses *Edge Detection* tidak menghilangkan fitur penting dari tangan.

3. Visualisasi Hasil Augmentasi Geometrik

```
# Visualisasi Augmentasi
def visualize_augmentation_grid(x_data, idx=0):
    sample_image = x_data[idx]
    sample_image = np.expand_dims(sample_image, axis=0)
    augmentations = {
        "Original": ImageDataGenerator(),
        "Rotation (15°)": ImageDataGenerator(rotation_range=15),
        "Width Shift (10%)": ImageDataGenerator(width_shift_range=0.1),
        "Height Shift (10%)": ImageDataGenerator(height_shift_range=0.1),
        "Shear (10%)": ImageDataGenerator(shear_range=0.1),
        "Zoom (20%)": ImageDataGenerator(zoom_range=[0.2, 1]),
        "Horizontal Flip": ImageDataGenerator(horizontal_flip=True),
    }
    fig, axes = plt.subplots(nrows=1, ncols=len(augmentations),
                           figsize=(len(augmentations) * 3, 4))

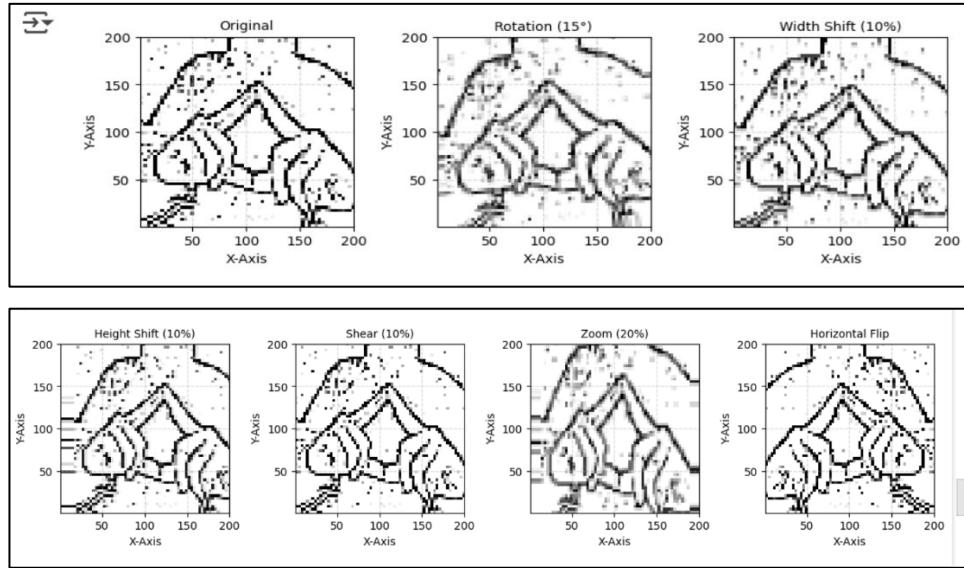
    for i, (aug_name, datagen) in enumerate(augmentations.items()):
        augmented_image = next(datagen.flow(sample_image, batch_size=1))
        # Menampilkan gambar dalam grafik koordinat dengan skala 1-200
        axes[i].imshow(augmented_image[0].squeeze(), cmap="gray", extent=[1,
        200, 1, 200])
        axes[i].set_title(aug_name, fontsize=10)
        axes[i].set_xlabel("X-Axis")
        axes[i].set_ylabel("Y-Axis")
        axes[i].grid(True, linestyle='--', alpha=0.5)

    plt.tight_layout()
    plt.show()

random_idx = random.randint(0, len(x_train) - 1)
visualize_augmentation_grid(x_train, idx=random_idx)
```

Gambar 4.18. Program Visualisasi Hasil Augmentasi Geometrik

Hasil visualisasi ini menunjukkan perbedaan transformasi augmentasi data yang diterapkan pada dataset. Dalam proses ini, augmentasi diterapkan pada objek dalam gambar menggunakan ImageDataGenerator, tanpa mengubah ukuran gambar secara keseluruhan. Augmentasi yang diterapkan meliputi rotasi, translasi, *shearing*, *scaling*, dan *flipping*. *Output* dari fungsi ini, yaitu:



Gambar 4.19. Visualisasi Hasil Augmentasi Geometrik

Berikut adalah analisis dari masing-masing transformasi yang diterapkan:

- Gambar pertama adalah gambar asli hasil *Edge Detection* tanpa perubahan. Ini digunakan sebagai acuan sebelum dilakukan augmentasi.
- Gambar diputar sebesar ± 15 derajat berdasarkan pusat gambar.
- Gambar digeser secara horizontal sebesar 10% dari lebar gambar dan gambar mengalami pergeseran vertikal sebesar 10% dari tinggi gambar.
- Gambar mengalami distorsi miring (*shearing*) dengan faktor 0.1.
- Gambar diperbesar hingga 20% dari ukuran aslinya.
- Gambar dicerminkan secara horizontal, menyebabkan tampilan tangan terbalik ke arah yang berlawanan.

Tabel 4.1. Efek Augmentasi Geometrik pada Gambar

Transformasi	Efek pada Gambar
Rotation (15°)	Memutar gambar ke kiri 15°
Translation (10%)	Menggeser gambar ke kanan dan sedikit ke atas
Scaling (20%)	Memperbesar gambar 20%
Shearing (10%)	Membuat gambar sedikit tampak miring
Flipping (Horizontal)	Membalik gambar ke arah horizontal

Berdasarkan hasil implementasi augmentasi menggunakan `ImageDataGenerator`, dapat disimpulkan bahwa transformasi dilakukan secara acak tetapi tetap mengikuti aturan yang telah ditentukan dalam fungsi. Setiap gambar mengalami perubahan sesuai dengan parameter yang diberikan, seperti rotasi dalam rentang tertentu, translasi berdasarkan persentase ukuran gambar, *scaling* untuk memperbesar atau memperkecil gambar, serta *shearing* untuk menciptakan efek kemiringan. Augmentasi dilakukan pada piksel objek yang ada dalam gambar. Proses-proses transformasi hanya menggeser atau mengubah bentuk objek, bukan ukuran gambar secara keseluruhan. `ImageDataGenerator` melakukan augmentasi secara *on-the-fly* terhadap objek tanpa mengubah ukuran gambar *input*. Meskipun nilai spesifik transformasi dipilih secara acak dalam setiap iterasi, augmentasi tetap berada dalam batas yang telah didefinisikan oleh penulis, sehingga memastikan keanekaragaman data tanpa keluar dari aturan yang ditentukan.

Namun, hasil augmentasi menunjukkan bahwa transformasi dengan parameter kecil terkadang kurang terlihat secara signifikan dalam gambar hasil augmentasi. Sebagai contoh, rotasi 15° atau translasi sebesar 10% masih menghasilkan perubahan yang relatif kecil, sehingga efeknya terhadap diversifikasi data tidak terlalu kuat. Oleh karena itu, untuk meningkatkan variasi data, parameter augmentasi dapat diperbesar, misalnya dengan rotasi hingga 20° atau 30° , translasi hingga 20%, atau *scaling* lebih besar agar perubahan lebih terlihat tanpa menghilangkan karakteristik utama gambar.

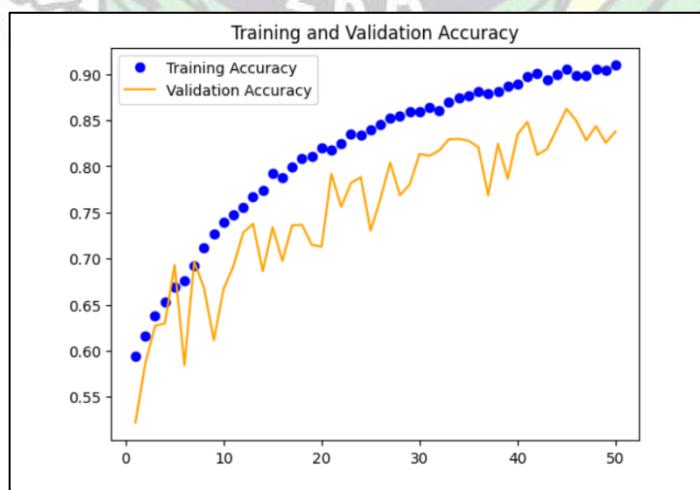
4.2.2. Pengujian Kinerja Model CNN

Pengujian ini digunakan untuk mengukur dan mengevaluasi kinerja model CNN setelah proses pelatihan selesai. Evaluasi dilakukan melalui visualisasi metrik pelatihan (akurasi dan *loss*) dan menghitung *test accuracy* menggunakan `test_generator`.

```
# Fungsi Plot Akurasi dan Loss
def plot_loss(epochs, loss, val_loss):
    plt.plot(epochs, loss, 'bo', label='Training Loss')
    plt.plot(epochs, val_loss, 'orange', label='Validation Loss')
    plt.title('Training and Validation Loss')
    plt.legend()
    plt.show()
def plot_accuracy(epochs, acc, val_acc):
    plt.plot(epochs, acc, 'bo', label='Training Accuracy')
    plt.plot(epochs, val_acc, 'orange', label='Validation Accuracy')
    plt.title('Training and Validation Accuracy')
    plt.legend()
    plt.show()
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
plot_loss(range(1, len(loss) + 1), loss, val_loss)
plot_accuracy(range(1, len(loss) + 1), acc, val_acc)
```

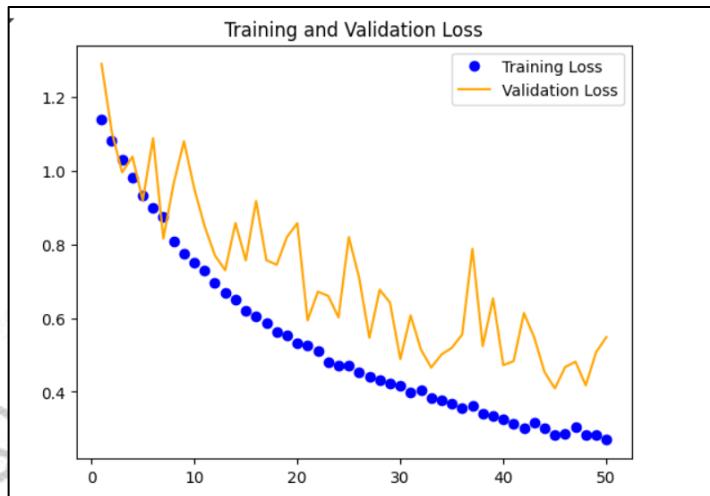
Gambar 4.20. Program Uji Model CNN

Fungsi `plot_loss()` digunakan untuk menampilkan grafik *loss* selama proses pelatihan, di mana "*Training Loss*" (kerugian model pada data latih) dibandingkan dengan "*Validation Loss*" (kerugian pada data validasi) dalam setiap *epoch*. Sementara itu, fungsi `plot_accuracy()` digunakan untuk menampilkan grafik akurasi pelatihan dan validasi, yang menggambarkan seberapa baik model mengenali pola dalam dataset.



Gambar 4.21. Grafik *Training* dan *Validation Accuracy*

Berdasarkan hasil *training* model CNN selama 50 *epoch*, model menunjukkan peningkatan akurasi yang signifikan dari awal hingga akhir pelatihan. Seiring bertambahnya *epoch*, *training accuracy* terus meningkat hingga mencapai 90.95%, sementara *validation accuracy* mencapai 83.77%. Grafik *Training* and *Validation Accuracy* menunjukkan tren kenaikan yang stabil, meskipun terdapat sedikit fluktuasi pada *validation accuracy*.



Gambar 4.22. Grafik *Training* dan *Validation Loss*

Grafik *Training* and *Validation Loss* juga memperlihatkan penurunan loss secara bertahap, baik pada *training loss* maupun *validation loss*. Seiring bertambahnya *epoch*, *training loss* terus menurun hingga 0.2690, sementara *validation loss* juga berkurang hingga 0.5481. Meskipun ada perbedaan antara *training loss* dan *validation loss* di beberapa titik, perbedaan ini masih dalam batas wajar, sehingga model tidak mengalami *overfitting* yang ekstrem.

Dari hasil ini, dapat disimpulkan bahwa model CNN telah berhasil mempelajari pola dari dataset dengan cukup baik. Namun, fluktuasi pada *validation loss* dan *validation accuracy* mengindikasikan bahwa model masih dapat ditingkatkan lebih lanjut dengan penyesuaian parameter augmentasi, peningkatan jumlah data latih, atau pemilihan arsitektur model yang lebih optimal.

Setelah visualisasi metrik pelatihan, model diuji menggunakan fungsi `model.evaluate()`, yang menghitung *test accuracy* menggunakan `test_generator`. Evaluasi ini dilakukan dalam 50 *batch* dari data uji untuk mengukur seberapa baik model mampu mengklasifikasikan gambar yang belum pernah dilihat sebelumnya.

Akurasi akhir model ditampilkan dalam format persentase, yang menunjukkan tingkat keberhasilan model dalam mengenali bahasa isyarat dari gambar hasil *pre-processing*.

```
# Evaluasi Model
test_loss, test_accuracy = model.evaluate(test_generator, steps=50)
print(f"Test accuracy: {test_accuracy * 100:.2f}%")
```

Gambar 4.23. Program Evaluasi Model CNN

Hasil dari `model.evaluate()` ini adalah persentase hasil *test accuracy* yang menunjukkan angka:

```
[ ] # Evaluasi Model
test_loss, test_accuracy = model.evaluate(test_generator, steps=50)
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")

→ 50/50 ━━━━━━ 4s 82ms/step - accuracy: 0.8420 - loss: 0.5194
Test Accuracy: 83.88%
```

Gambar 4.24. Hasil Evaluasi Model CNN

Berdasarkan hasil evaluasi model CNN menggunakan data uji (*test set*), diperoleh *test accuracy* sebesar 83.88% dan *test loss* sebesar 0.5194. Nilai akurasi ini menunjukkan bahwa model mampu mengklasifikasikan data uji dengan tingkat keberhasilan yang cukup baik, dengan artian model telah belajar dari data *training* dan dapat menggeneralisasi pola yang telah dipelajari ke data yang belum pernah dilihat sebelumnya.

Namun, nilai *test loss* sebesar 0.5194 mengindikasikan bahwa meskipun model memiliki akurasi yang tinggi, masih terdapat kesalahan dalam prediksi, yang mungkin disebabkan oleh beberapa faktor seperti variasi dalam dataset uji, kompleksitas data yang tidak sepenuhnya tertangkap oleh model, atau masih adanya *noise* dalam data setelah proses *pre-processing*.

BAB 5

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Hasil yang diperoleh berdasarkan penelitian terhadap penerapan Augmentasi Geometrik dan *Edge Detection* sebagai bagian dari *pre-processing* pada BISINDO dalam menghadapi variasi gerakan secara *real-time*, yaitu:

1. *Edge Detection* yang diterapkan terlebih dahulu berhasil mengekstraksi fitur utama, menyoroti batas tangan dan bentuk jari secara jelas dengan menggunakan *Gaussian Blur*, *Adaptive Thresholding*, dan *Otsu's Thresholding*.
2. Augmentasi Geometrik berupa rotasi, translasi, *scaling*, *shearing*, dan *flipping* yang diterapkan berhasil memperbanyak dataset menjadi 380.800 data gambar dengan visualisasi yang menunjukkan bahwa augmentasi tetap mempertahankan bentuk dasar dari isyarat tangan.
3. Model CNN yang digunakan mencapai akurasi validasi sebesar 83.77% dan akurasi pengujian sebesar 83.88%, dimana menunjukkan bahwa *pre-processing* yang diterapkan mampu membantu model dalam meningkatkan akurasi klasifikasi.
4. Beberapa parameter augmentasi memiliki efek yang kurang signifikan dengan hasil pengujian menunjukkan bahwa parameter rotasi 15° dan translasi 10% masih kurang menghasilkan variasi yang signifikan dalam dataset.

Berdasarkan dari *point-point* diatas, dapat disimpulkan bahwa penerapan Augmentasi Geometrik dan *Edge Detection* pada tahap *pre-processing* dataset efektif dalam meningkatkan akurasi model dalam mengenali gestur BISINDO.

5.2. Saran

Adapun saran untuk pengembangan penelitian selanjutnya adalah:

1. Diharapkan pada penelitian selanjutnya dapat meningkatkan parameter augmentasi agar hasil lebih terlihat dan transformasi data lebih bervariasi.
2. Diharapkan agar penelitian selanjutnya dapat menggunakan arsitektur

model pendekripsi yang lebih dalam atau penambahan fitur seperti *dropout* dan *batch normalization* untuk meningkatkan akurasi, terutama untuk diterapkan dalam aplikasi *video call real-time*.

3. Disarankan untuk melakukan studi lebih lanjut mengenai teknik augmentasi selain `ImageDataGenerator` yang dapat digunakan untuk meningkatkan kualitas dataset dan menghasilkan augmentasi data yang lebih jelas.



DAFTAR PUSTAKA

- Al-Rahlawee, A. T. H., dan Rahebi, J. 2021. Multilevel Thresholding of images with improved Otsu Thresholding by black widow optimization algorithm. *Springer*. 80:28217–28243.
- Alshdaifat, E., Alshdaifat, D., Alsarhan, A., Hussein, F., dan El-Sahli, S. M. F. S. 2021. The effect of preprocessing techniques applied to numeric features on classification algorithms' performance. *Data*. 6(2):11.
- Awaluddin, B. A., Chao, C. T., dan Chiou, J. S. 2023. Investigating effective geometric transformation for image augmentation to improve static hand gestures with a pre-trained Convolutional Neural Network. *Mathematics*. 11(23):4783.
- Badan Pusat Statistik. 2022. Jumlah penyandang tuli di Indonesia. *Badan Pusat Statistik Nasional, Jakarta*.
- Bergstrom, A. C., Conran, D., dan Messinger, D. W. 2023. Gaussian blur and relative edge response. *arXiv*.
- De la Rosa, F. L., Gomez-Sirvent, J. L., Sanchez-Reolid, R., Morales, R., dan Fernandez-Caballero, A. 2022. Geometric transformation-based data augmentation on defect classification of segmented images of semiconductor materials using a ResNet50 Convolutional Neural Network. *Elsevier*. 206:117731.
- Devi, T. G., Patil, N., Rai, S., dan Philipose, C. S. 2023. Gaussian blurring technique for detecting and classifying acute lymphoblastic leukemia cancer cells from microscopic biopsy images. *Life*. 13(2):348.
- Dong, S., Wang, P., dan Abbas, K. 2021. A survey on Deep Learning and its applications. *Computer Science Review*. 40:100379.
- Elgendi, M., Nasir, M. U., Tang, Q., Smith, D., Grenier, J., Batte, C., Spieler, B., Leslie, W. D., Menon, C., Fletcher, R. R., Howard, N., Ward, R., Parker, W., dan Nicolaou, S. 2021. The effectiveness of image augmentation in Deep Learning networks for detecting COVID-19: A geometric transformation perspective. *Frontiers in Medicine*. 8.

- Fadillah, R. Z., Irawan, A., dan Susanty, M. 2021. Data augmentasi untuk mengatasi keterbatasan data pada model penerjemah bahasa isyarat Indonesia (BISINDO). *Jurnal Informatika*. 8(2):208-214.
- Hikmatia, N. A. E., dan Zul, M. I. 2021. Aplikasi penerjemah bahasa isyarat Indonesia menjadi suara berbasis Android menggunakan Tensorflow. *Jurnal Politeknik Caltex Riau*. 7(1):74-83.
- Janiesch, C., Zschech, P., dan Heinrich, K. 2021. Machine learning and Deep Learning. *Springer*. 31:685-695.
- Jardim, S., Antonio, J., dan Mora, C. 2023. Image Thresholding approaches for medical image segmentation - short literature review. *Procedia Computer Science*. pp. 1485–1492.
- Kim, M., Seifert, R., Fragemann, J., Kersting, D., Murray, J., Jonske, F., Pomykala, K. L., Egger, J., Fendler, W. P., Hermann, K., dan Kleesiek, J. 2023. Evaluation of Thresholding methods for the quantification of [68Ga] Ga PSMA 11 PET molecular tumor volume and their effect on survival prediction in patients with advanced prostate cancer undergoing [177Lu]Lu PSMA 617 radioligand therapy. *European Journal of Nuclear Medicine and Molecular Imaging*. 50:2196–2209.
- Korn, O., Holt, R., Kontopoulos, E., Kappers, A. M. L., Persson, N. K., dan Olson, N. 2018. Empowering persons with deafblindness: designing an intelligent assistive wearable in the SUITCEYES project. *Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference (PETRA'18)*. New York, NY: Association for Computing Machinery, pp. 545–551.
- Lubis, M. H., dan Sani, A. 2014. Analisis kualitas video call menggunakan perangkat NSN Flexi Packet Radio. *Singuda ENSIKOM*. 6(2):76-80.
- Maharana, K., Mondal, S., dan Nemade, B. 2022. A review: Data Pre-processing and data augmentation techniques. *Global Transitions Proceedings*. 3(1):91-99.
- Mumuni, A., dan Mumuni, F. 2022. Data augmentation: A comprehensive survey of modern approaches. *Elsevier*. 16:100258.

- Nalbant, K. G., dan Uyanik, S. 2021. Computer Vision in the metaverse. *Journal of Metaverse*. 1(1):9-12.
- Normalisa, Rachmaniar, A., Diana, D., Saefudin, M., dan Parulian, R. 2022. Application of Computer Vision detection of apples and oranges using Python language. *Journal of Information System, Informatics and Computing*. 6(2):455-466.
- Pratiwi, N. I. 2017. Penggunaan media video call dalam teknologi komunikasi. *Jurnal Ilmiah Dinamika Sosial*. 1(2):202-224.
- Pu, M., Huang, Y., Liu, Y., Guan, Q., dan Ling, H. 2022. EDTER: Edge Detection with Transformer. *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1402-1412.
- Rizky, A. D. B., Faqihuddin, M. A., Romadhan, F. F., dan Siradjuddin, I. A. 2023. Identifikasi alfabet bahasa isyarat Indonesia dengan menggunakan Convolutional LSTM. *Seminar Nasional Sinergitas Era Digital 5.0 dalam Pembangunan Teknologi Hijau Berkelanjutan*. 9 Desember 2023, Malang, Indonesia, pp. 183-190.
- Su, Z., Liu, W., Yu, Z., Hu, D., Liao, Q., Tian, Q., Pietikainen, M., dan Liu, L. 2021. Pixel difference networks for efficient Edge Detection. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5117-5127.
- Taye, M. M. 2023. Theoretical understanding of Convolutional Neural Network: Concepts, architectures, applications, future directions. *Computation*. 11(3):52.
- Winnemoller, H., Kyprianidis, J. E., dan Olsen, S. C. 2012. XDoG: An extended difference-of-Gaussians compendium including advanced image stylization. *Computers & Graphics*. 36(6):720-753.

Lampiran : 3 (tiga) lembar
Hal : Permohonan Persetujuan Dosen Wali Akademik
untuk Bentuk Lain Setara Skripsi

Yth. Ibu Fuzy Yustika Manik S.Kom., M.Kom
Dosen Pembimbing Akademik
Program Studi S-1 Ilmu Komputer
Fakultas Ilmu Komputer dan Teknologi Informasi
Universitas Sumatera Utara
Medan

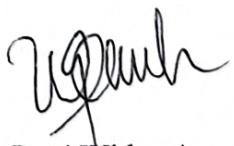
Dengan hormat, berdasarkan Keputusan Kepala Balai Pengembangan Talenta Indonesia Pusat Prestasi Nasional Sekretariat Jenderal Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi Nomor 1653/J7.1/PN.00/2024 tentang Penetapan Pemenang Pekan Ilmiah Mahasiswa Nasional (PIMNAS) Ke 37 Tahun 2024, dimana saya dan tim lolos sebagai Pemenang Medali Emas kategori Poster bidang PKM-Karsa Cipta pada Pekan Ilmiah Mahasiswa Nasional (PIMNAS) Ke 37 Tahun 2024. Oleh karena itu saya memohon untuk dapat mengajukannya sebagai Bentuk Lain Setara Skripsi untuk penyelesaian Tugas Akhir saya. Adapun judul yang akan saya ajukan sebagai berikut:

“Penerapan Teknik Augmentasi Geometrik dan *Edge Detection* Pada *Pre-processing* Dataset Bahasa Isyarat Indonesia (BISINDO) untuk Mengatasi Variasi Gerakan *Real-time* dalam Aplikasi ElCue”

Bersama surat ini saya lampirkan dokumen pendukung untuk pengajuan Bentuk lain Setara Skripsi.

Demikian permohonan ini disampaikan, atas perhatian Ibu diucapkan terima kasih.

Medan, 13 Maret 2025
Hormat saya,



Rani Widya Astuti
NIM. 211401018

**SURAT PERSETUJUAN DOSEN WALI AKADEMIK
TENTANG BENTUK LAIN SETARA SKRIPSI**

Saya yang bertanda tangan di bawah ini:

Nama : Fuzy Yustika Manik S.Kom., M.Kom
NIP : 198710152019032010
Pangkat/Golongan : Lektor/IIIc
Program Studi : S-1 Ilmu Komputer
Fakultas : Ilmu Komputer dan Teknologi Informasi - USU

Menyetujui permohonan untuk mengajukan Bentuk Lain Setara Skripsi untuk memenuhi Tugas Akhir sebagai syarat kelulusan mahasiswa sebagai berikut:

Nama : Rani Widya Astuti
NIM : 211401018
Program Studi : S-1 Ilmu Komputer
Bentuk Lain Setara Skripsi : Karya Ilmiah yang dimenangkan pada Pekan Ilmiah Nasional (PIMNAS) ke-37 Tahun 2024 (Medali Emas)
Judul : Penerapan Teknik Augmentasi Geometrik dan *Edge Detection* Pada *Pre-processing* Dataset Bahasa Isyarat Indonesia (BISINDO) untuk Mengatasi Variasi Gerakan *Real-time* dalam Aplikasi ElCue

Demikian disampaikan agar dapat dipergunakan dengan sebaiknya.

Medan, 13 Maret 2025

Dosen Pembimbing Akademik,


Fuzy Yustika Manik S.Kom., M.Kom
NIP 198710152019032010



**Universitas Sumatera Utara
Fakultas Ilmu Komputer dan
Teknologi Informasi**

Alamat
Jalan Universitas
No. 9 Kampus USU,
Medan 20155

Email:
fasilkomti@usu.ac.id
Telepon: (061) 8213793

**SURAT KEPUTUSAN
DEKAN FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
NOMOR: 1532/UN5.2.14.D/SK/HK.07/2025**

Tentang

Susunan Personalia Dosen Pembimbing Tugas Akhir Mahasiswa Program Studi (S-1) Ilmu Komputer
Fakultas Ilmu Komputer dan Teknologi Informasi (Fasilkom-TI) Universitas Sumatera Utara

Dekan Fakultas Ilmu Komputer dan Teknologi Informasi USU

- Menimbang** : Bahwa Tugas Akhir adalah karya ilmiah tertulis dan/atau prototipe, atau projek, baik secara individu maupun berkelompok sebagai syarat penyelesaian studi akademik, maka dipandang perlu untuk menetapkan Dosen Pembimbing Penyusunan Tugas Akhir mahasiswa (i) yang bersangkutan.
- Mengingat** : 1. Undang-undang Nomor 20 tahun 2003 tentang Sistem Pendidikan Nasional;
2. Peraturan Pemerintah Nomor: 48 tahun 1957 tentang Penetapan Pendirian USU;
3. Peraturan Pemerintah Nomor: 19 tahun 2005 tentang Standar Nasional Pendidikan;
4. Peraturan Pemerintah Nomor: 17 tahun 2010 tentang Pengelolaan dan Penyelenggaraan Pendidikan;
5. Keputusan Rektor USU;
a. Nomor: 03/UN5.1.R/SK/SPB/2021 tentang Peraturan Akademik Program Sarjana (S1) USU;
c. Nomor: 1876/UN5.1.R/SK/SDM/2021 tentang Pengangkatan Dekan Fasilkom-TI USU Periode 2021-2026.
6. Keputusan Rektor Nomor 459/UN5.1.R/SK/SPB/2023 tentang Pedoman Pelaksanaan Program Merdeka Belajar Universitas Sumatera Utara.
7. Keputusan Dekan Nomor 2968/UN5.2.1.14/SPB/2023 tentang Pedoman Rekognisi Prestasi Akademik dan Non-Akademik Kegiatan Mahasiswa dan Pelaksanaan Program Merdeka Belajar.
- Membaca** : Hasil persetujuan Dosen Wali Akademik tentang Bentuk Lain Setara Skripsi mahasiswa yang bersangkutan oleh Fakultas Ilmu Komputer dan Teknologi Informasi USU tanggal 20 Maret 2025 dengan judul :
- "Penerapan Teknik Augmentasi Geometrik dan Edge Detection pada Pre-processing Dataset Bahasa Isyarat Indonesia (BISINDO) untuk Mengatasi Variasi Gerakan Real-time dalam Aplikasi ElCue"
- Memutuskan**
- Menetapkan** : Susunan Personalia Pembimbing Tugas Akhir Seorang Mahasiswa Program Studi (S-1) Ilmu Komputer Fasilkom TI USU Medan sebagai berikut:
1. Mahasiswa terbimbing adalah :
Nama : Rani Widya Astuti
NIM : 211401018
Program Studi : S-1 Ilmu Komputer Fasilkom-TI USU
 2. Dosen Pembimbing:
1. Hayatunnufus, S.Kom., M.Cs. (Pembimbing Pertama)
NIP 199207192024062001
 2. Fuzy Yustika Manik, S.Kom., M.Kom. (Pembimbing Kedua)
NIP 198710152019032010
 3. Surat Keputusan ini berlaku selama 6 (enam) bulan sejak tanggal **21 Maret 2025 sampai dengan 21 September 2025**. Apabila mahasiswa belum menyelesaikan tugas akhir dalam waktu tersebut, maka Surat Keputusan ini dapat dievaluasi kembali.
 4. Segala sesuatu akan diperbaiki kembali, jika di kemudian hari ternyata terdapat kekeliruan dalam Surat Keputusan ini.

Medan, 24 Maret 2025

Ditandatangani secara elektronik oleh:
Dekan



Tembusan:

1. Dosen Pembimbing Tugas Akhir
2. Yang bersangkutan

**Maya Silvi Lydia
NIP 197401272002122001**

**SK KEPALA BPTI, PUSPRESNAS, KEMENDIKBUD RISTEK NOMOR
1653/J7.1/PN.00/2024**

No	Kelas	Ket	Nama	Perguruan Tinggi	Judul
23.	PKM-PM4	Perak	KHAIRUNNISA' NUR FADHILAH	Universitas Negeri Surabaya	Metasekerta: Action-Adventure Game Bertema Kesenian Wayang Wong untuk Mengoptimalkan Culture-Based Learning di SMP PGRI 1 Buduran Sidoarjo
24.	PKM-PM4	Perunggu	AULIA NURUL HIKMAH	Institut Pertanian Bogor	Rintik Muda: Pengaruh Self-Control dengan Metode Ant-Fection sebagai Upaya Pengembangan Diri pada Anak Yayasan Titipan Ummat Indonesia
25.	PKM-KC1	Emas	MUCHAMMAD ADAM WILDAN	Universitas Airlangga	AUVEST: Automatic Vest with Cooling and Heating System Terintegrasi Aplikasi Mobile Sebagai Upaya Pencegahan Heat Stress Pada Pekerja Konstruksi
26.	PKM-KC1	Perak	BURHANUDIN YUSUF ABDULLAH AR RAMADHAN	Universitas Negeri Malang	Desain dan Implementasi Sistem Teknologi Berbantuan Virtual Reality dan Exoskeleton sebagai Perangkat Rehabilitasi Pasca Stroke
27.	PKM-KC1	Perunggu	SETIYAKI ARUMA NANDI	Universitas Brawijaya	Sistem Rekomendasi Tanaman Pertanian Berbasis Remote Sensing dan Machine Learning dengan Data Citra Satelit Sentinel-2A Berdasarkan Karakteristik Tanah
28.	PKM-KC2	Emas	FRENGKI PRABOWO SAPUTRO WIJAYANTO	Universitas Gadjah Mada	EMO-vest: Futuristic Vest as Epilepsy Detector with Airguard Protecting Based on IoT
29.	PKM-KC2	Perak	KRISNA SEIYA EKIAWAN	Universitas Brawijaya	Alat Deteksi Dini Bladder Cancer Berbasis Quantum Dots dengan Fluorescence Resonance Energy Transfer Terintegrasi Application Programming Interface Klasifikasi Fuzzy Logic
30.	PKM-KC2	Perunggu	MOCH. AVIN	Institut Teknologi Sepuluh Nopember	Integrated Otoscope Examination melalui Klasifikasi Citra Tympanic Membrane untuk Deteksi Otitis Media sebagai Preventif Conductive Hearing Loss berbasis Deep Learning
31.	PKM-KC3	Emas	TESSA AGITHA IRWANI BR BARUS	Universitas Sumatera Utara	EICue : Aplikasi Video Call dengan Pendekripsi Bahasa Isyarat Indonesia (BISINDO) Berbasis Computer Vision dan Deep Learning untuk Meningkatkan Komunikasi Inklusif
32.	PKM-KC3	Perak	RAISA ZAHRA SALASABILA	Politeknik Negeri Madiun	Sistem Identifikasi dan Monitoring Potensi Bahaya dan Kelalaian Perilaku Pekerja Mesin Gerinda Berbasis Deep Learning Upaya Mencegah Kecelakaan Kerja
33.	PKM-KC3	Perunggu	WAHYU PUTRA	Universitas	Smart Vision (SmarV): Alat Pendekripsi Obat secara Real-Time Berbasis Algoritma Deep

DOKUMENTASI PENGUMUMAN PIMNAS KE-37



SERTIFIKAT PESERTA DAN PEMENANG PIMNAS KE-37



**FLYER/POSTER LOMBA PEKAN ILMIAH MAHASISWA NASIONAL (PIMNAS)
KE – 37**



PENJELASAN PERANAN MAHASISWA DALAM TIM

No	Nama / NIM	Program Studi	Uraian Tugas
1	Tessa Agitha Irwani Br. Barus / 211401138	S-1 Ilmu Komputer	<ul style="list-style-type: none"> Bertugas melakukan <i>fine-tuning</i> pada model MobileNetV2. Mendesain dan menguji tambahan layer (<i>custom layers</i>) untuk meningkatkan akurasi deteksi. Mengukur performa model (akurasi, <i>latency</i>) pada input <i>real-time</i>.
2	Putri Andriyani / 211401008	S-1 Ilmu Komputer	<ul style="list-style-type: none"> Membangun aplikasi ElCue dengan Flutter. Mengintegrasikan layanan <i>backend</i> (Firebase) dan <i>video call</i> (Agora). Menyambungkan model deteksi BISINDO ke tampilan aplikasi secara <i>real-time</i>.
3	Rani Widya Astuti / 211401018	S-1 Ilmu Komputer	<ul style="list-style-type: none"> Meneliti dan menerapkan teknik <i>pre-processing</i> berupa augmentasi data dan <i>edge detection</i>. Memastikan model tetap <i>robust</i> terhadap variasi gerakan tangan dalam kondisi <i>real-time</i>. Bekerja sama membuat model untuk memastikan hasil <i>pre-processing</i> optimal.
4	Angela Siadari / 211401030	S-1 Ilmu Komputer	<ul style="list-style-type: none"> Merancang dan mengembangkan sistem pengumpulan dataset BISINDO (video & gambar). Bertanggung jawab atas labeling, anotasi data, dan validasi kualitas data. Mengembangkan <i>pipeline</i> inferensi berbasis <i>Computer Vision</i> untuk integrasi ke sistem ElCue.
5	Helga Pricilla Br. Purba / 211401067	S-1 Ilmu Komputer	<ul style="list-style-type: none"> Mengintegrasikan hasil dari model <i>deep learning</i> dan <i>speech recognition</i> ke sistem ElCue. Mengatur alur komunikasi antar input <i>gesture</i> dan suara untuk meningkatkan komunikasi 2 arah. Menjamin sinkronisasi dan performa dari dua jalur input (<i>gesture + voice</i>).



PRIMARY SOURCES

1	Submitted to Universitas Sumatera Utara Student Paper	2%
2	repository.usu.ac.id Internet Source	2%
3	123dok.com Internet Source	1%
4	www.frontiersin.org Internet Source	1%
5	text-id.123dok.com Internet Source	1%
6	www.mdpi.com Internet Source	<1%
7	biomed.reviewer.ly Internet Source	<1%
8	link.springer.com Internet Source	<1%
9	digilib.uinsa.ac.id Internet Source	<1%
10	repository.its.ac.id Internet Source	<1%
11	www.coursehero.com Internet Source	<1%
12	adoc.pub Internet Source	<1%