IMPLEMENTASI MODEL YOLOv11 DENGAN MEDIAPIPE PADA SISTEM PENERJEMAH BAHASA ISYARAT INDONESIA (SIBI) UNTUK MEDIA PEMBELAJARAN

(Skripsi)

Oleh

ZAKI AHMAD BASYARY
NPM 2215061004

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNVERISTAS LAMPUNG
BANDAR LAMPUNG
2026

IMPLEMENTASI MODEL YOLOv11 DENGAN MEDIAPIPE PADA SISTEM PENERJEMAH BAHASA ISYARAT INDONESIA (SIBI) UNTUK MEDIA PEMBELAJARAN

Oleh

ZAKI AHMAD BASYARY

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA TEKNIK

Pada

Jurusan Teknik Elektro
Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2026

ABSTRAK

Implementasi Model YOLOv11 Dengan Mediapipe pada Sistem Penerjemah Bahasa Isyarat Indonesia (SIBI) Untuk Media Pembelajaran

Oleh

Zaki Ahmad Basyary

Pengembangan sistem penerjemah Bahasa Isyarat Indonesia (SIBI) secara real-time masih menghadapi tantangan dalam menyeimbangkan akurasi dan kecepatan pemrosesan. Untuk mengatasi hal tersebut, penelitian ini merancang sistem penerjemah SIBI sebagai media pembelajaran dengan mengimplementasikan algoritma YOLOv11 dan MediaPipe pada dataset citra tangan statis yang terdiri dari 1.847 gambar dalam 35 kelas. Penelitian ini juga melakukan analisis perbandingan pada beberapa varian model YOLOv11 untuk menentukan model yang paling optimal. Hasil penelitian menunjukkan bahwa sistem mampu mencapai akurasi di atas 90% dengan kecepatan lebih dari 15 FPS. Model YOLOv11 Nano memberikan performa terbaik dengan mAP@0.5 sebesar 0,94655 (training) dan 0,936 (testing), serta mAP@0.5:0.95 sebesar 0,77426 (training) dan 0,776 (testing). Pengujian real-time pada perangkat berbasis CPU menunjukkan akurasi hingga 97% dengan kecepatan 18,63 FPS. Dengan demikian, YOLOv11 Nano menjadi model paling optimal karena memiliki keseimbangan terbaik antara akurasi dan efisiensi komputasi untuk implementasi sistem penerjemah SIBI secara real-time.

Kata Kunci: SIBI, YOLOv11, MediaPipe, *Real-time*

ABSTRACT

Implementation of the YOLOv11 Model with Mediapipe on the Indonesian Sign Language Interpreter System (SIBI) for Learning Media

By

Zaki Ahmad Basyary

The development of a real-time Indonesian Sign Language (SIBI) translation system still faces challenges in balancing accuracy and processing speed. To address this, this study designed a SIBI translation system as a learning medium by implementing the YOLOv11 algorithm and MediaPipe on a static hand image dataset consisting of 1,847 images in 35 classes. This study also compared several YOLOv11 model variants to determine the most optimal model. The results showed that the system was able to achieve accuracy above 90% at a speed of more than 15 FPS. The YOLOv11 Nano model provided the best performance with mAP@0.5 of 0.94655 (training) and 0.936 (testing), and mAP@0.5:0.95 of 0.77426 (training) and 0.776 (testing). Real-time testing on a CPU-based device showed an accuracy of up to 97% at a speed of 18.63 FPS. Thus, YOLOv11 Nano is the most optimal model because it has the best balance between accuracy and computational efficiency for the implementation of a real-time SIBI translator system.

Keywords: *SIBI, YOLOv11, MediaPipe, Real-time*

Judul Skripsi	:	IMPLEMENTASI MODEL YOLOV11 DENGAN MEDIAPIPE PADA SISTEM PENERJEMAH BAHASA ISYARAT INDONESIA (SIBI) UNTUK MEDIA PEMBELAJARAN
Nama Mahasiswa	:	Zaki Ahmad Basyary
Nomor Pokok Mahasiswa	:	2215061004
Program Studi	:	S1 Teknik Informatika
Jurusan	:	Teknik Elektro
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Yessi Mulyani, S.T., M.T.	**Puput Budi Wintoro, S. Kom, M.T.I.**
NIP. 197312262000122001	NIP. 198410312019031004

2. **Mengetahui**

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T., M.T.	**Yessi Mulyani, S.T., M.T.**
NIP. 197103141999032001	NIP. 197312262000122001

MENGESAHKAN

1. Tim Penguji

Ketua : **Yessi Mulyani S.T., M.T.** .....

Sekretaris : **Puput Budi Wintoro, S. Kom, M.T.I.** .....

Penguji : **Dr. Ir. M. Komarudin S.T., M.T.** .....

2. Dekan Fakultas Teknik

Dr. Ahmad Herison, S.T., M.T

NIP. 196910302000031001

Tanggal Lulus Ujian Skripsi : **3 Februari 2026**

SURAT PERNYATAAN

Saya yang bertanda tangan dibawah ini, menyatakan dengan sebenarnya bahwa skripsi saya yang berjudul " Implementasi Model Yolov11 Dengan Mediapipe Pada Sistem Penerjemah Bahasa Isyarat Indonesia (Sibi) Untuk Media Pembelajaran " merupakan hasil karya saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi saya ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 18 April 2026

Pembuat Pernyataan

Zaki Ahmad Basyary

2215061004

RIWAYAT HIDUP

Penulis dilahirkan di Lampung Selatan pada tanggal 5 Juli 2004, sebagai anak kedua dari pasangan Bapak Akhmad Syarifudin dan Ibu Tuti Amriyani. Penulis memulai pendidikan di TK Mekar Sari pada tahun 2009, kemudian melanjutkan ke SD Negeri 3 Negara Ratu dan lulus pada tahun 2016. Pendidikan menengah pertama diselesaikan di SMP Negeri 1 Natar pada tahun 2018, dan pendidikan menengah atas di SMA Negeri 1 Natar yang diselesaikan pada tahun 2022. Selanjutnya, penulis melanjutkan pendidikan tinggi di Universitas Lampung, Program Studi S1 Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik, melalui jalur SNMPTN (Seleksi Nasional Masuk Perguruan Tinggi Negeri). Selama menjalani masa perkuliahan, penulis aktif berpartisipasi dalam berbagai kegiatan, antara lain:

1. Berperan sebagai Asisten Praktikum sekaligus Koordinator di Laboratorium Teknik Komputer Universitas Lampung pada tahun 2024 hingga 2026.
2. Menjadi anggota Himpunan Mahasiswa Teknik Elektro (HIMATRO) pada periode 2023 dan periode 2024.
3. Mengikuti program Studi Independen Bersertifikat (MSIB) dari Kementerian Pendidikan dan Kebudayaan, sebagai Data Analyst di Organisasi MIKTI (Masyarakat Industri Kreatif Teknologi Informasi dan Komunikasi Indonesia) pada tahun 2024.
4. Mengikuti program magang yang diselenggarakan oleh Kementerian Keuangan Republik Indonesia pada Direktorat Jenderal Perbendaharaan Provinsi Lampung, di bidang PPA II (Pembinaan Pelaksanaan Anggaran II) sebagai Data Analyst pada tahun 2025.
5. Berperan sebagai *Team Leader Data analyst* pada program *Internship* yang diselengarakan oleh startup yaitu SOKO Financial pada tahun 2025.

MOTTO

وَاَنْ لَّيْسَ لِلْاِنْسَانِ اِلَّا مَا سَعٰىۙ

“Bahwa manusia hanya memperoleh apa yang telah diusahakannya”

(QS. An-Najm · Ayat 39)

“Hasil adalah cerminan dari usaha.”

(Pepatah)

PERSEMBAHAN

Bismillahirrahmanirrahim

Puji dan syukur saya panjatkan ke hadirat Allah SWT, Tuhan Yang Maha Kuasa, atas segala rahmat, taufik, dan hidayah-Nya sehingga saya sebagai penulis dapat menyelesaikan skripsi ini dengan baik. Shalawat serta salam semoga senantiasa tercurah kepada Nabi Muhammad SAW, beserta keluarga, sahabat, dan seluruh umatnya, yang telah membawa umat manusia dari zaman kegelapan menuju zaman yang penuh ilmu pengetahuan dan cahaya kebenaran.

Kupersembahkan karya ini kepada:

Kedua orang tua tercinta dan kakak tersayang, yang senantiasa memberikan doa, dukungan, kasih sayang, serta semangat tanpa henti demi keberhasilan penulis dalam menyelesaikan karya tulis ini. Terima kasih atas segala pengorbanan dan ketulusan yang telah diberikan. Semoga Allah SWT senantiasa melimpahkan kesehatan, keberkahan, dan perlindungan di dunia maupun di akhirat.

Sahabat dan teman-teman seperjuangan yang selalu memberikan motivasi, bantuan, serta kebersamaan selama proses perkuliahan hingga penyusunan skripsi ini.Seluruh dosen, staf, dan civitas akademika Program Studi Teknik Informatika, Jurusan Teknik Elektro, Universitas Lampung, yang telah memberikan ilmu, arahan, serta bimbingan kepada penulis selama masa studi.

SANWACANA

Puji Syukur penulis ucapkan ke hadirat Allah SWT, Tuhan Yang Maha Esa, atas segala rahmat, berkah, dan hidayah-Nya yang senantiasa dilimpahkan, terutama nikmat kesehatan, kesempatan, serta waktu. Berkat rahmat dan ridha-Nya, penulis akhirnya dapat menyelesaikan skripsi yang berjudul "Implementasi Model YOLOv11 dengan MediaPipe pada Sistem Penerjemah Bahasa Isyarat Indonesia (SIBI) untuk Media Pembelajaran." Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Program Studi S1 Teknik Informatika, Fakultas Teknik, Universitas Lampung. Proses penyusunan skripsi ini berlangsung selama kurang lebih 3bulan dengan berbekal ilmu, pengalaman, serta pembelajaran yang telah penulis tempuh selama masa perkuliahan.

Penulis menyadari bahwa penyelesaian skripsi ini tidak terlepas dari dukungan bimbingan, dan motivasi dari berbagai pihak. Oleh karena itu, dengan tulus hati, penulis mengucapkan terimakasih kepada:

1. Allah SWT, Tuhan Yang Maha Esa, atas segala limpahan Rahmat dan hidayah-Nya.
2. Kedua orang tua yaitu Bapak Akhmad Syarifudin dan Ibu Tuti Amriyani yang sel. Kedua orang tua tercinta yaitu Bapak Akhmad Syarifudin dan Ibu Tuti Amriyani serta seluruh keluarga yang senantiasa memberikan doa, kasih sayang, dukungan moral maupun materi, serta semangat yang tiada henti kepada penulis.
3. Ibu Yessi Mulyani, S.T., M.T., selaku Dosen Pembimbing I, yang telah meluangkan waktu, tenaga, dan pikiran untuk memberikan arahan, bimbingan, serta masukan yang sangat berarti kepada penulis selama proses penyusunan skripsi ini.
4. Bapak Puput Budi Wintoro, S.Kom., M.T.I., selaku Dosen Pembimbing II sekaligus Dosen Pembimbing Akademik, yang telah memberikan arahan,

bimbingan, serta nasihat akademik kepada penulis selama masa perkuliahan dan proses penyusunan skripsi ini.

5. Bapak Dr. Ir. M. Komarudin, S.T., M.T., selaku Dosen Penguji, yang telah memberikan kritik, saran, serta masukan yang membangun demi penyempurnaan skripsi ini sehingga menjadi lebih baik dan sistematis.
6. SLB Dharma Bhakti Dharma Pertiwi beserta seluruh guru dan siswa yang telah memberikan kesempatan, dukungan, serta bantuan kepada penulis dalam proses pelaksanaan penelitian.
7. Teman-teman tim proyek yang telah bekerja sama, berbagi ide, serta saling mendukung selama proses penelitian dan penyusunan skripsi ini.
8. Teman-teman Asisten Laboratorium Teknik Komputer angkatan 2021, 2022, dan 2023 yang telah memberikan dukungan, kebersamaan, serta pengalaman berharga selama proses perkuliahan dan penyusunan skripsi ini.
9. Seluruh teman-teman Program Studi Teknik Informatika serta civitas akademika yang telah memberikan dukungan, kebersamaan, dan kontribusi selama masa perkuliahan hingga penyusunan skripsi ini.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun demi perbaikan di masa mendatang. Semoga skripsi ini dapat memberikan manfaat dan kontribusi bagi pengembangan ilmu pengetahuan, khususnya di bidang Teknik Informatika, serta menjadi referensi yang bermanfaat bagi penelitian selanjutnya.

Bandar Lampung, 18 April 2026

Penulis

Zaki Ahmad Basyary

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR TABEL

Tabel Halaman

1. PENDAHULUAN

1.1 Latar Belakang

Komunikasi merupakan aspek penting dalam kehidupan manusia untuk saling bertukar informasi, gagasan, dan perasaan. Bagi penyandang disabilitas rungu, komunikasi verbal menjadi tantangan utama karena keterbatasan dalam mendengar dan berbicara. Berdasarkan data dari Kementerian Sosial Republik Indonesia tahun 2024 menyatakan bahwa sekitar total 1% dari populasi Masyarakat Indonesia mengalami gangguan pendengaran yaitu sekitar lebih 2 juta jiwa [1]. Bahasa isyarat digunakan sebagai sarana utama dalam berkomunikasi sehari-hari. Tetapi tidak semua orang baik itu penyandang disabilitas maupun masyarakat umum memiliki kesempatan untuk mempelajari bahasa isyarat dengan baik dengan praktik karena keterbatasan media pembelajaran, serta minimnya akses terhadap edukasi [2].

Bahasa isyarat yang digunakan di Indonesia terbagi menjadi 2 jenis yaitu Sistem Isyarat Indonesia (SIBI) dan Bahasa Isyarat Indonesia (BISINDO) [3]. SIBI dikembangkan berdasarkan aturan struktural bahasa Indonesia dan mengadopsi beberapa bentuk dari bahasa isyarat internasional. Sedangkan BISINDO merupakan bahasa isyarat konseptual yang tumbuh secara alami dalam komunitas Tuli dan digunakan untuk komunikasi sehari-hari. SIBI yang menjadi bahasa isyarat yang lebih umum dipakai di Sekolah Luar Biasa (SLB) karena mengikuti tata bahasa Indonesia dan sesuai dengan kurikulum pendidikan formal Oleh karena itu, pada penelitian ini lebih menekankan penggunaan SIBI dalam pembuatan system praktik dalam media pembelajaran.

Media pembelajaran dikelompokkan menjadi empat jenis, yaitu: (a) media hasil teknologi cetak, (b) media hasil teknologi audiovisual, (c) media hasil teknologi berbasis komputer, dan (d) media hasil gabungan teknologi cetak dan computer [4].

Beberapa penelitian terdahulu menggunakan media teknologi berbasis computer, seperti penggunaan motion graphic, media visual interaktif, dan aplikasi digital [5]. Salah satu media yang digunakan yaitu menggunakan teknologi AI untuk praktik pengenalan bahasa isyarat [6]. Oleh karena itu, sistem ini menggunakan pengembangan model kecerdasan buatan yang mampu mengenali bahasa isyarat melalui pemrosesan citra dari input kamera, sehingga pengguna dapat mempraktikkan gerakan Bahasa Isyarat Indonesia (SIBI) secara langsung dan memperoleh hasil penerjemahan secara real-time.

Dalam beberapa tahun terakhir, berbagai penelitian telah dilakukan untuk mengembangkan sistem penerjemah bahasa isyarat berbasis *computer vision* guna untuk pengenalan bahasa isyarat. Sistem ini umumnya menggunakan pengolahan citra dan pembelajaran mesin (*Machine Learning*) maupun *deep learning* untuk mengenali pola gerakan tangan dan menerjemahkannya menjadi teks. Tetapi sebagian sistem yang ada masih memiliki keterbatasan, seperti akurasi yang rendah ketika digunakan pada kondisi pencahayaan berbeda, atau belum mampu mendeteksi gerakan secara *Real-time* [7], [8]. Masalah tersebut yang menjadi tantangan dalam pembuatan model penerjemah bahasa isyarat tersebut.

Beberapa penelitian sebelumnya yang menggunakan metode *Convolutional Neural network (CNN)* sering membutuhkan waktu komputasi tinggi dan kurang efisien ketika diterapkan pada perangkat dengan spesifikasi terbatas [9]. Dan terdapat penelitian yang mengatasi keterbatasan akurasi dan efisiensi sistem penerjemah bahasa isyarat dengan memodifikasi arsitektur CNN. Beberapa penelitian menambahkan custom layer, meningkatkan jumlah *feature map*s, atau memanfaatkan kombinasi *Convolution* dan *Pooling* yang lebih dalam untuk meningkatkan kemampuan ekstraksi fitur [10], [11]. Selain itu, pendekatan transfer learning menggunakan model seperti MobileNet, VGG, dan ResNet juga pernah digunakan untuk mempercepat proses pelatihan sekaligus meningkatkan performa model [12], [13]. Model-model tersebut memang mampu menghasilkan akurasi yang baik, namun sebagian besar belum diuji secara menyeluruh pada skenario *real-time* sehingga performanya dalam kondisi penggunaan langsung masih belum dapat dipastikan.

Berdasarkan permasalahan tersebut, penelitian ini dilakukan secara tim dengan mengusulkan pengembangan sistem penerjemah Bahasa Isyarat Indonesia (SIBI) secara real-time sebagai media pembelajaran. Dalam penelitian ini, fokus pada pengembang model kecerdasan buatan, yang memanfaatkan kombinasi MediaPipe dan model YOLOv11 untuk melakukan proses pengenalan dan penerjemahan bahasa isyarat. MediaPipe dipilih karena mampu mendeteksi titik-titik penting (landmark) tangan secara cepat dan efisien[11]. Serta YOLOv11 sebagai model deteksi model terbaru dengan performa *real-time* yang unggul dengan arsitektur yang lebih ringan dan efisien[14]. Dengan menggabungkan keunggulan kedua metode tersebut, sistem yang diusulkan diharapkan mampu bekerja dengan cepat, akurat, dan tetap ringan sehingga dapat diimplementasikan pada perangkat dengan kemampuan komputasi menengah seperti laptop maupun smartphone.

1.2 Rumusan Masalah

Adapun rumusan masalah dari penelitian ini, yaitu sebagai berikut.

1. Bagaimana mengimplementasikan YOLOv11 dan MediaPipe agar menghasilkan performa optimal dalam pendeteksian bahasa isyarat SIBI, meliputi akurasi yang baik dan kecepatan yang memadai dengan komputasi yang efisien?
2. Bagaimana hasil perbandingan kinerja beberapa versi model YOLOv11 dengan MediaPipe dalam mengenali bahasa isyarat SIBI berdasarkan metrik evaluasi?
3. Seberapa baik performa model YOLOv11 dalam pengujian *real-time* untuk mengenali bahasa isyarat SIBI berdasarkan akurasi dan kecepatan pemrosesan?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah sebagai berikut:

1. Mengimplementasikan kombinasi YOLOv11 dan MediaPipe untuk menghasilkan performa optimal dalam pendeteksian bahasa isyarat SIBI, meliputi akurasi yang baik dan kecepatan yang memadai dengan komputasi yang efisien.

2. Menganalisis dan membandingkan kinerja beberapa versi model YOLOv11 yang dikombinasikan dengan MediaPipe dalam mengenali bahasa isyarat SIBI berdasarkan metrik evaluasi yang relevan.
3. Menganalisis kinerja model YOLOv11 saat melakukan pengujian dalam mengenali dan menerjemahkan isyarat tangan secara *Real-time* berdasarkan akurasi dan kecepatan pemrosesan.

1.4 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Menyediakan sarana membantu proses penerjemahan bahasa isyarat SIBI ke dalam bentuk teks secara otomatis, sehingga dapat mendukung kegiatan belajar dan praktik dalam media pembelajaran.
2. Memberikan kontribusi dalam pengembangan sistem penerjemahan bahasa isyarat SIBI berbasis *Deep Learning* yang mampu bekerja secara *real-time*.
3. Menjadi referensi pengembangan sistem penerjemahan bahasa isyarat yang lebih efisien dan *Real-time.*

1.5 Batasan Penelitian

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Penelitian ini hanya berfokus pada penerjemahan isyarat tangan SIBI untuk media pembelajaran berupa huruf alfabet dan beberapa kata dasar yang umum digunakan dalam komunikasi sehari-hari.
2. Sistem ini dikembangkan sebagai media pembelajaran yang hanya menyediakan penerjemahan satu arah dari Bahasa Isyarat Indonesia (SIBI) ke teks, tanpa mencakup penerjemahan dari teks atau suara ke bahasa isyarat.
3. Penelitian ini difokuskan pada pengujian sistem yang berjalan pada perangkat dengan spesifikasi rendah (*low device*), khususnya perangkat berbasis CPU.
4. Dataset yang digunakan merupakan dataset citra tangan statis, dari sumber *public*, tanpa melibatkan deteksi gerakan dinamis atau kalimat beruntun.

5. Sistem yang dikembangkan hanya menampilkan hasil penerjemahan dalam bentuk teks, tanpa output suara.

1.6 Sistematika Penulisan Skripsi

Adapun sistematika penulisan dari penelitian ini sebagai berikut :

BAB I : PENDAHULUAN

Bab ini membahas membahas mengenai latar belakang penelitian, perumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Bagian bab ini memaparkan teori-teori yang terkait dengan penelitian serta sebagai penunjang penelitian.

BAB III : METODE PENELITIAN

Bagian bab ini membahas mengenai waktu dan tempat penelitan, jadwal penelitian, alat penelitian, serta metode penelitian yang digunakan dalam pengembangan Aplikasi

BAB IV : HASIL DAN PEMBAHASAN

Bagian bab ini membahas mengenai hasil dan pembahasan sesuai dengan tahapan-tahapan penelitian.

BAB V : KESIMPULAN DAN SARAN

Bagian bab ini membahas mengenai kesimpulan berdasarkan dari hasil penelitian, serta saran untuk penelitian lebih lanjut.

DAFTAR PUSTAKA : Bab ini memuat daftar sumber kutipan teori yang dijadikan acuan dalam penulisan laporan.

II. TINJAUAN PUSTAKA

2.1 Bahasa Isyarat

Bahasa adalah sebuah sarana yang digunakan untuk komunikasi yang merupakan kemampuan yang didapat dari kebiasaan yang dilakukan sehari-hari dalam menyampaian informasi kepada orang lain [15]. Sedangkan komunikasi adalah suatu cara yang digunakan untuk mempermudah dalam menyampaikan suatu informasi melalui bentuk verbal maupun nonverbal [16]. Komunikasi dapat terjadi ketika 2 orang memiliki bahasa yang sama sehingga dapat dimengerti. Oleh karena itu, terdapat beberapa macam bahasa yang digunakan oleh setiap orang didunia. Salah satu bahasa yang digunakan oleh orang yang memiliki kebutuhan khusus seperti tuna rungu berkomunikasi dengan menggunakan bahasa isyarat.

Bahasa isyarat adalah bahasa yang menggunakan gerakan dari anggota tubuh seperti tangan, ekspresi wajah untuk berkomunikasi dan menyampikan suatu informasi kepada orang lain [17]. Bahasa isyarat berbeda dengan bahasa lisan yang dimana perbedaannya terletak pada sarana produksi dan persepsinya. Bahasa lisan di produksi melalui organ artikulatoris dan di persepsi dengan alat pendengaran yaitu telingga sedangkan bahasa Isyarat diproduksi melalui gestur seperti tangan dan ekspresi serta dipersepsikan dengan indra penglihatan [18].

Bahasa isyarat Indonesia merupakan bahasa isyarat yang digunakan oleh Masyarakat Indonesia untuk berkomunikasi. Sistem isyarat Bahasa Indonesia merupakan tatanan sistematik yang menggunakan jari, tangan, dan gerakan lainnya untuk menggambarkan kosakata yang akan disampaikan [17]. Di Indonesia memiliki 2 jenis bahasa isyarat yang sering digunakan oleh Masyarakat Indonesia. Bahasa isyarat tersebut yaitu sebagai berikut :

2.1.1 Sistem Isyarat Bahasa Indonesia (SIBI)

Sistem isyarat Bahasa Indonesia atau yang disingkat SIBI merupakan salah satu jenis dari bahasa isyarat di Indonesia yang digunakan dalam berkomunikasi sesame kaum tunarungu dan tunawicara pada masyarakat lebih luas [19]. Menurut Kementerian Sosial Republik Indonesia tahun 2021, menyatakan bahwasanya SIBI digunakan pada kurikulum Pendidikan Indonesia yang diajarkan pada Sekolah Luas Biasa sebagai bahasa yang digunakan. Keputusan Menteri Pendidikan dan Kebudayaan Republik Indonesia Nomor 0161/U/2994, Pemerintah Indonesia membakukan Kamus Sistem Isyarat Bahasa Indonesia menggunakan SIBI sehingga SIBI digunakan dalam Pendidikan yang diajarkan di SLB dibawah naungan Kementrian Pendidikan dan Kebudayaan Indonesia.

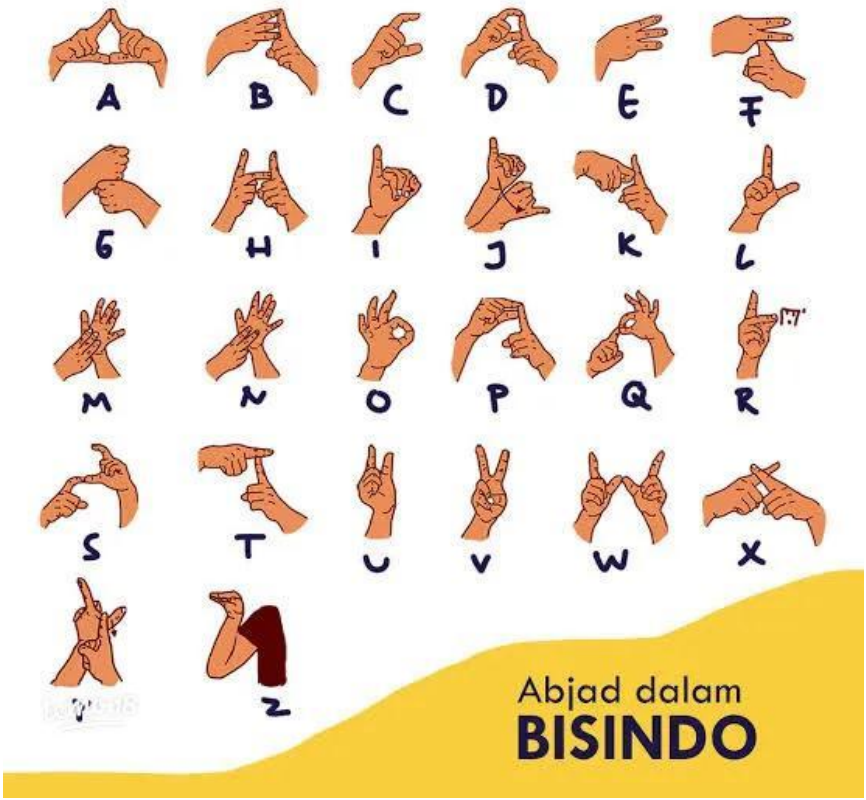
Gambar 1. Sistem Isyarat Bahasa Indonesia (SIBI)

Pada SIBI lebih banyak menggunakan 1 tangan daripada 2 tangan seperti pada Gambar 1. Dalam pengembangannya, SIBI dikembangkan bukan dari orang tuli melainkan dari orang normal yang dikembangkan dari bahasa isyarat *America Sign Language (ASL)* [17].

2.1.2 Bahasa Isyarat Indonesia (BISINDO)

Bahasa isyarat Indonesia (BISINDO) merupakan bahasa isyarat yang banyak digunakan dalam kehidupan sehari hari karena tidak terlalu komplek dan juga

BISINDO dianggap sebagai bahasa ibu bagi orang tuli di Indonesia sehingga kebanyakan orang-orang Indonesia menggunakan BISINDO sebagai bahasa sehari-hari [17]. Berbeda dengan SIBI yang menekankan struktur yang kompleks, BISINDO lebih menekankan pemanfaatan gerakan tangan, ekspresi wajah dalam menyampaikan informasi. Sehingg BISINDO lebih banyak menggunakan 2 tangan dalam penggunannya.



Gambar 2. Bahasa Isyarat Indonesia (BISINDO)

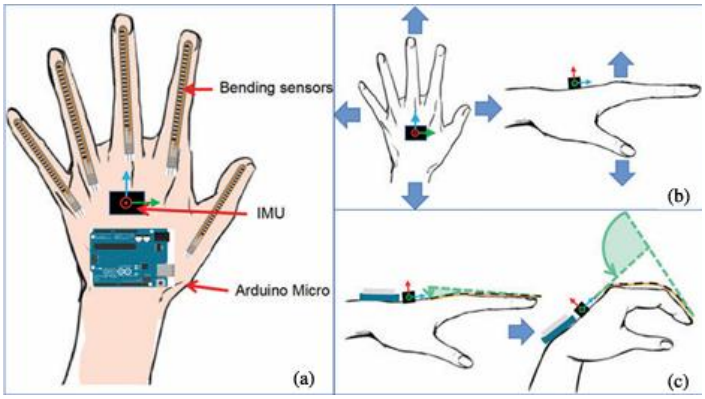
2.2 *Sign Language Recognition* (SLR)

Sign Language Recoginition merupakan topik penelitian yang mencakup berbagai bidang penelitian yaitu meliputi *object detection, trajectory tracking, pose estimation, action recognition,* dan bidang lain terkait [7]. Sign Language Recognition menunjuk pada proses otomatis dalam menerjemahkan suatu gerakan yang dalam hal ini gerakan tangan atau gerakan tubuh (isyarat) menggunakan teknologi *Computer Vision* dan *Deep Learning* [20]. SLR digunakan untuk

membantu komunikasi antara tunarungu dengan orang normal agar dapat berkomunikasi satu sama lain dengan lebih mudah.

2.2.1 Teknologi *Sign Language Recognition*

Dalam perkembangannya, sign language recognition menggunakan beberapa teknologi, teknologi yang digunakan meliputi *sensor-based* dan *vision-based.* Pada *sensor-based,* teknologi yang digunakan dalam melakukan SLR yaitu menggunakan sensor [20]. Sensor ini digunakan untuk mengukur dan merekam data terkait pergerakan tangan termasuk pembekokan, bentuk, perpindahan posisi yang memiliki makna dalam bahasa isyarat. Sensor yang digunakan seperti sensor regangan, sensor elektromiografi permukaan (sEMG), dan sensor lainnya. Sensor tersebut diletakkan dalam suatu sarung tangan sehingga dapat mengatasi ketidaknyamanan dalam menggunakan sensor. Sarung tangan tersebut disebut *Sign-Glove*. *Sign-Glove* merupakan sarung tangan yang diletakkan beberapa sensor yang berguna dalam pengenalan bahasa isyarat seperti sensor yang mengukur bentuk dan perpindahan tangan.



Gambar 3. Glove-Sign

Penggunaan *Glove-Sign* memiliki kelebihan yaitu meminimalkan penggunaan penglihatan dalam penerjemahan bahasa isyarat lalu meminimalkan penggunaan daya pemrosesan. Tetapi teknologi *Glove-sign* memiliki kelemahan yaitu mengharuskan penggunaan sarung tangan yang cukup rumit, sehingga dapat membatasi gerakan pada orang yang menggunakannya [20]. Oleh karena itu munculnya teknologi yang kedua yaitu *vision-based.*

Teknologi *vision-based* merujuk pada penggunaan kamera atau perangkat yang menyediakan penangkapan gambar. Dengan penggunaan tersebut menghilangkan kebutuhan untuk menempelkan sensor pada tubuh seperti pada *sensor-based* [20]. Dalam menggunakan *vision-based* akan menghasilkan gambar atau video yang akan dilakukan *preprocessing* untuk mengekstraksi fitur-fitur yang ada lalu dilakukan pembuatan model AI dalam melakukan penerjemahan bahasa isyarat tersebut. Tetapi pada teknologi *vision-based* memiliki tantangan pada faktor lingkungan seperti pencahayaan yang berbeda, sudut pandang, orientasi, dan juga kerumitan dalam gerakan yang semakin sulit untuk dilakukan penerjemahan.

2.2.2 Jenis-Jenis *Sign Language Recognition*

Sign Language Recognition (SLR) memiliki 2 pendekatan utama yaitu : *isolated sign recognition* (ISLR) dan *Continuous Sign Language Recognition* (CSLR) [8]. *Isolated sign recognition* atau pengenalan bahasa isyarat terisolasi adalah pengenalan bahasa isyarat yang berfokus pada pengenalan isyarat terpisah (kata Tunggal). Tujuannya adalah mengenali setiap isyarat secara akurat agar dapat digunakan sebagai dasar untuk proses penerjemahan bahasa isyarat ke bentuk tulisan atau ucapan. Sementara itu, *Continuous Sign Language Recognition* (CSLR) berfungsi sebagai tahap lanjutan yang lebih kompleks. Pendekatan ini mengenali rangkaian isyarat secara berurutan dalam satu kalimat atau frasa, sehingga lebih mencerminkan komunikasi alami pengguna bahasa isyarat. CSLR bertujuan menjaga makna dan konteks antar isyarat, meskipun prosesnya lebih rumit dibanding ISLR.

2.3 *Artificial Intelligence*

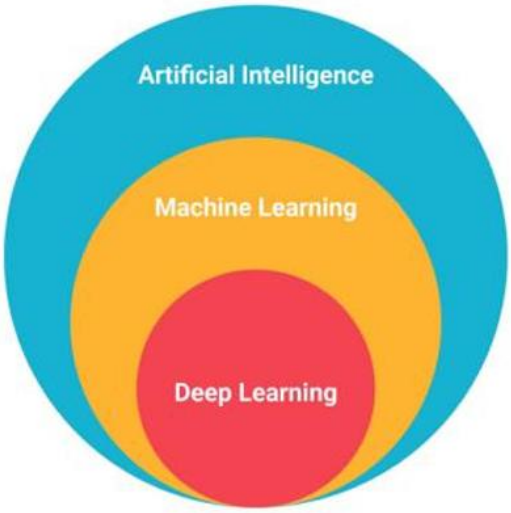
Kecerdasan buatan (*Artificial Intelligence* atau *AI*) merupakan bidang ilmu yang berfokus pada pengembangan sistem atau mesin yang mampu meniru perilaku dan kecerdasan manusia [21]. Sistem ini dirancang untuk dapat membantu manusia dalam menyelesaikan masalah dengan cara membuat mesin dapat bukan hanya berfikir tetapi berperilaku seperti manusia. Kecerdasan buatan atau *Artificial Intelligence* memiliki arti dalam bahasa latin yaitu “intelligo” yang berarti “saya paham” yang memiliki makna yaitu handal dalam mengerti dan juga melaksanakan

aksi [22]. AI mulai berkembang pada tahun 1940-an dan mulai dalam masa aktif pengembangan yaitu pada tahun 1950.

Definisi AI dikategorikan menjadi 4 kategori yaitu *thingking humanly, acting humanly, thinking rationally, dan acting rationally* [23]. sebagai berikut :

1. *Thingking humanly,* ini berarti AI dapat berpikir seperti manusia dengan cara meniru dalam proses berpikir manusia.
2. *Acting humanly,* ini berarti AI bukan hanya pada cara berpikir, tetapi juga pada hasil tindakan yang menyerupai tindakan manusia.
3. *Thinking rationally,* ini berarti AI menekankan dalam berpikir berdasarkan logika dan penalaran dalam mengambil keputusan bukan hanya meniru berpikir manusia tetapi melihat dari aturan logis yang tepat.
4. *Acting rationally,* ini berarti AI sudah dapat berpikir secara rasional /logis dan juga dapat bertindak secara rasional berdasarkan informasi logis tersebut. AI tidak hanya berpikir logis, tetapi juga menyesuaikan tindakannya terhadap kondisi nyata agar hasilnya optimal.

AI atau *Artificial Intelligence* merupakan bidang ilmu yang memiliki beberapa cabang ilmu pengetahuan didalamnya yaitu *Machine Learning* dan *Deep Learning*.



Gambar 4. *Artificial Intelligence, Machine Learning, dan Deep Learning*

2.4 *Machine Learning*

Machine Learning merupakan cabang ilmu disiplin ilmu dari *Artificial Intelligence* yang membahas terkait pengembangan system berbasiskan data [21]. *Machine Learning* berarti mesin atau system yang belajar seperti cara belajar manusia. *Machine Learning* belajar dari data yang ada untuk dianalisis pola untuk menemukan suatu informasi atau insight. Pengembangan *Machine Learning* menerapkan beberapa ilmu didalamnya yaitu meliputi fisika, statisitka, matematika, dan data mining sehingga mesin mampu belajar suatu analisa tanpa diprogram kembali.

Machine Learning (pembelajaran mesin) merupakan kumpulan metode yang memungkinkan sistem untuk secara otomatis mengenali pola dalam data, kemudian memanfaatkan pola tersebut untuk memprediksi data di masa mendatang atau membantu pengambilan keputusan dalam kondisi yang tidak pasti [24]. Saat ini *Machine Learning* sudah digunakan dibeberapa bidang untuk membantu pekerjaan manusia.

Berdasarkan jenis tipe pembelajarannya, *Machine Learning* dikategorikan menjadi beberapa kategori yaitu supervised learning, unsupervised learning, semi-supervised leaning, dan reinforcement leaning [24]. Berikut penjelasannya:

1. *Supervised learning* atau yang berarti pembelajaran terarah merupakan tipe pembelajaran *Machine Learning* yang dimana data yang digunakan sudah memiliki label sehingga tujuannnya adalah mengelompokkan data ke dalam kelompok yang sudah ada (sudah berlabel) [21]. Tipe pembelajaran ini memerlukan data yang sudah berlabel yang digunakan untuk pelatihan (*training*) sehingga model dapat belajar melalui pelatihan tersebut. Pada umumnya, tipe pembelajaran ini digunakan untuk klasifikasi dan juga regresi [24]. Klasifikasi meliputi proses untuk memprediksi kelas atau label dari data baru yang tidak berlabel, sedangkan regresi memprediksi nilai output berdasarkan beberapa fitur masukan yang diperoleh dari data.
2. *UnSupervised learning* atau pembelajaran tak terarah merupakan kebalikan dari *Supervised learning* yang dimana data yang digunakan tidak memiliki label sehingga *Machine Learning* akan mendeteksi pola lalu mengelompokkan data

tersebut berdasarkan pola yang didapatkan [24]. Cara kerja dari tipe pembelejaran ini yaitu system akan mencari pola tersembunyi dari data yang tidak berlabel lalu menentukan korelasi antar data sehingga dapat dilakukan pengelompokan dari data tersebut. Salah satu contoh penerapannya yaitu untuk *Clustering* [21]. *Clustering* yaitu membuat model untuk mengelompokkan data input yang tak berlabel dikelompokan ke dalam beberapa kelas berdasarkan pola yang ditemukan.

3. *Semi-Supervised learning* atau pembelajaran semi-terarah merupakan gabungan dari tipe pembelajaran *Supervised learning* dan *unSupervised learning* yang dimana digunakan data yang berlabel dan juga tidak berlabel untuk menghasilkan output yang diinginkan [24]. Tipe pembelajaran ini hadir untuk mengatasi masalah data label yang sulit didapat tetapi data yang tidak berlabel tersedia dalam jumlah banyak. Dengan menggabungkan kedua data tersebut dapat menghasilkan prediksi yang akurat meskipun informasi label yang tersedia terbatas.
4. *Reinforcement learning* merupakan tipe pembelajaran *Machine Learning* yang dimana system AI akan mengamati lingkungan, kemudian melakukan tindakan dan terdapat balasan dari tindakan tersebut [21]. Tipe pembelajaran ini seperti bayi yang dimana akan belajar dengan sendiri berdasarkan balasan yang diberikan. Jika AI bertindak tidak sesuai dengan yang diinginkan maka akan ada hukuman, dan sebaliknya jika AI bertindak sesuai dengan yang diingkan maka akan ada hadiah. Dengan adanya hukuman dan hadiah maka AI akan belajar dengan sendirinya.

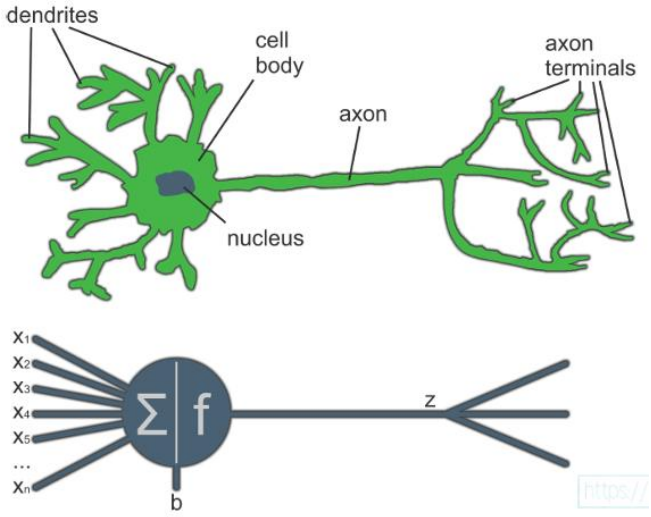
2.5 *Deep Learning*

Deep Learning (DL) merupakkan sub-bidang dari machine learing yang dimana mengimplementasikan jariangan saraf yang lebih dalam dan berlapis-lapis dalam melakukan analisa dan prediksi dibeberapa tugas seperti, computer vision, natural language processing, dan lainnya [21]. DL mengatasi dalam membuat model untuk data yang besar sehingga dibutuhkan algoritma yang dapat mengolah data besar tersebut. DL muncul karena ada kemajuan dalam bidang teknologi dan *Machine Learning* yaitu meliputi perangkat keras, kumpulan data, dan algoritma yang maju

[24]. Dengan adanya perangkat keras GPU yang mendukung pemrosesan DL yang jauh lebih cepat dibandingkan dengan CPU, dan dengan kebutuhan data yang semakin banyak sehingga diperlukan algoritma DL untuk pembelajaran yang lebih dalam. *Deep Learning* umumnya digunakan ketika ukuran data yang tersedia sangat besar, karena pada kondisi tersebut teknik ini mampu memberikan hasil yang lebih unggul dibandingkan metode *Machine Learning* tradisional. Tetapi dalam kondisi lain jika jumlah data terbatas, algoritma *Machine Learning* konvensional biasanya lebih efektif. Proses pelatihan model *Deep Learning* juga membutuhkan infrastruktur komputasi yang kuat agar dapat dilakukan dalam waktu yang efisien.

2.6 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (JST) atau artificial *neural network* merupakan cabang dari *Machine Learning* yang dimana merupakan jaringan komputasi yang terinspirasi dari jaringan saraf manusia [25]. Otak manusia terdiri dari jutaan neuron yang saling terhubung dan mampu membentuk pola atau aturan melalui pengalaman, sehingga dapat melakukan tugas-tugas kompleks seperti pengenalan pola, persepsi, dan pengendalian gerak jauh lebih cepat daripada komputer digital konvensional. jaringan syaraf tiruan dirancang untuk meniru mekanisme kerja otak tersebut dalam bentuk sistem komputasi yang terdiri atas unit-unit pemrosesan sederhana (*neurons* atau *processing units*) yang saling terhubung.

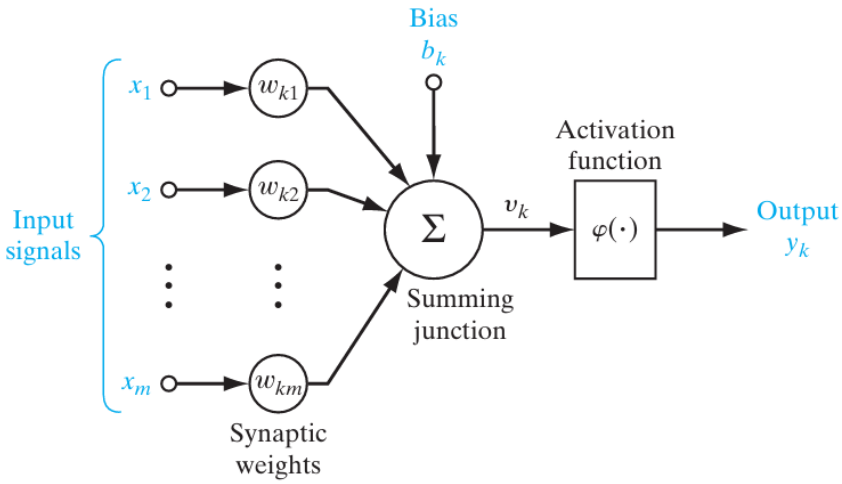


Gambar 5. Perbandingan Neruon Manusia dan Neuron Buatan

JST disebut menyerupai cara kerja otak manusia dalam 2 hal yaitu berupa pengetahuan yang diperoleh dari belajar melalui jaringan dan dalam hal kekuatan hubungan antar jaringan sel saraf yang disebut bobot sinaptik berperan sebagai media penyimpanan hasil pembelajaran [23]. Pendekatan ini menjadikan jaringan syaraf tiruan sebagai salah satu model adaptif yang efektif dalam berbagai bidang, seperti pengenalan pola, pengolahan citra, prediksi, serta sistem pengambilan keputusan berbasis pembelajaran.

2.6.1 Model Jaringan Syaraf Tiruan

Jaringan syaraf tiruan terdiri dari neuron buatan yang terhubung satu sama lainnya/ Neuron meruapkan unit pemrosesan informasi yang memiliki elemen dasarnya yaitu sinapsis, suatu adder, dan fungsi aktivasi [23]. Setiap neuron menerima sejumlah input, mengalikan masing-masing input tersebut dengan bobot (*weight*), menjumlahkannya, menambahkan bias, kemudian memproses hasilnya menggunakan suatu fungsi aktivasi *(activation function*) untuk menghasilkan output.



Gambar 6. Komponen Utama JST

Berikut ini penjelasan komponen dasar dalam jaringan syaraf tiruan yaitu sebagai berikut:

1. Neuron merupakan unit dasar dalam jaringan syaraf tiruan yang berfungsi untuk memproses informasi.

2. Bobot (*weight*) meruapakan parameter yang disesuaikan selama proses pelatihan agar model mampu melakukan prediksi yang akurat
3. Bias sebagai nilai tambahan yang membantu model menyesuaikan hasil komputasi agar tidak hanya bergantung pada nilai input yang dikalikan dengan bobot
4. Fungsi aktivasi digunakan untuk menentukan apakah sebuah neuron akan diaktifkan atau tidak, serta mengubah hasil perhitungan linier menjadi bentuk nonlinier.

Rumus sederhana dari perhitungan neuron yaitu sebagai berikut

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \quad (1)$$

Dengan keterangan yaitu :

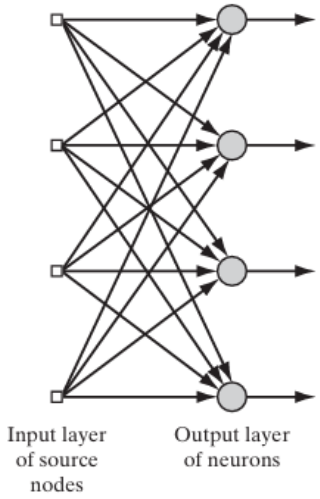
1. y = output
2. f = fungsi aktivasi yang digunakan
3. x_i = input ke-i
4. w_i = bobot (*weight*) untuk input ke-i
5. n = jumlah neuron
6. b = bias

2.6.2 Arsitektur Jaringan Syaraf Tiruan

Berikut ini merupakan arsitektur jaringan syaraf tiruan yang sering digunakan yaitu sebagai berikut :

A. *Single-layer Neural Network*

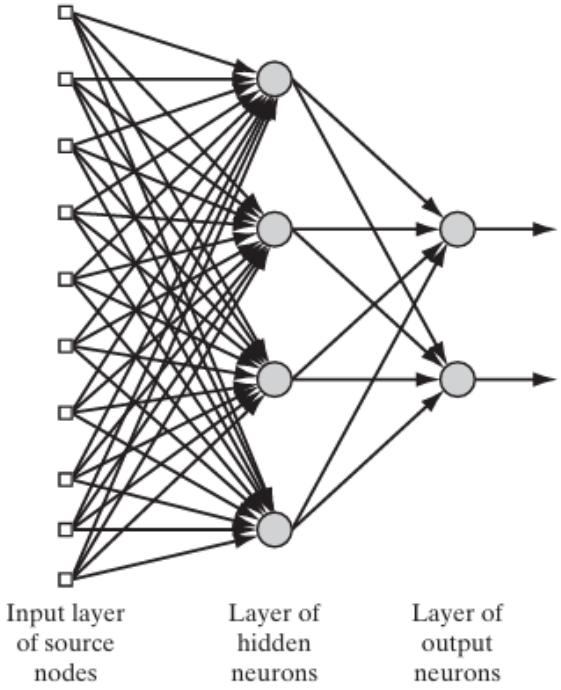
Single-layer network merupakan jaringan saraf tiruan dengan lapisan sederhana yang terdiri dari input layer dan output layer [23]. Berikut ini contoh pada gambar 2.7 yang meruapkan *Single-layer neural network* dengan 1 layer input dengan 4 node dan 1 layer output dengan 4 node.



Gambar 7. *Single-layer Neural Network*

B. *Multi-layer Neural Network*

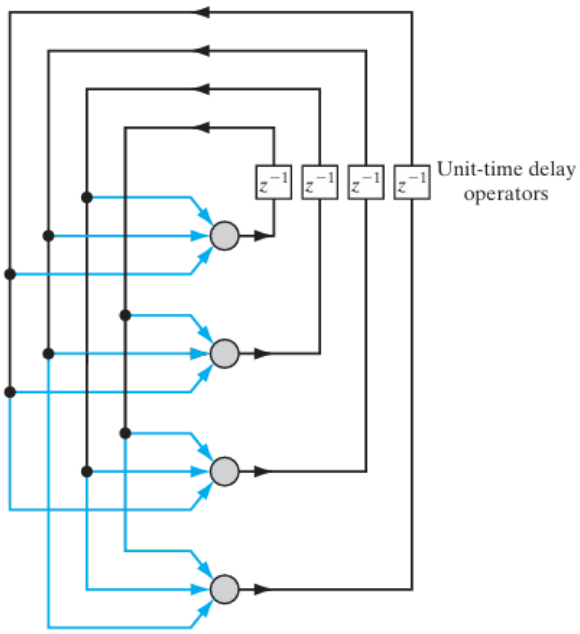
Multi-layer neural networks merupakan arsitektur jaringan syaraf tiruan yang bukan hanya terdiri dari lapisan input dan output tetapi memiliki lapisan tersembunyi (*hidden layer*) yang menguhubungkan antara lapisan input dengan outpt. Berikut ini contoh dari *multi-layer neural network* pada gambar 2.8 dengan 1 layer input yang memiliki 10 node neuron, lalu 1 lapisan tersembungin yang memiliki 4 node neuron, dan 1 lapisan output yang memiliki 2 node neuron.



Gambar 8. *Multi-layer Neural Network*

C. *Recurrent Networks*

Rucurrent network merupakan jaringan yang mempunyai minimal satu feedback loop. Dalam jaringan feedforward, sinyal hanya mengalir dari input ke output satu arah. Sedangkan pada RNN, output dari neuron dapat dikirim kembali ke input neuron lain (atau bahkan dirinya sendiri, jika ada *self-feedback*). Struktur ini membuat RNN mampu mengingat informasi dari waktu sebelumnya (memori jangka pendek).



Gambar 9. *Rucurrent Neural Network*

2.6.3 Cara Belajar Jaringan Syaraf Tiruan

Dalam proses pembelajaran atau *training* terdiri dari 2 jenis yaitu sebagai berikut:

A. *Forward Propagation*

Umpan maju atau forward propagation merupakan tahap awal dalam proses pembelajaran jaringan saraf tiruan. Pada tahap ini, data masukan (input data) dikirimkan melewati setiap neuron dari lapisan masukan menuju lapisan tersembunyi (*hidden layer*), hingga akhirnya mencapai lapisan keluaran (output layer) [22]. Setiap neuron pada lapisan tersembunyi melakukan proses perhitungan berdasarkan bobot (*weight*), bias, serta fungsi aktivasi yang digunakan. Fungsi aktivasi, seperti *Rectified Linear Unit* (ReLU), berperan penting dalam memperkenalkan non-li*near*itas agar jaringan dapat mempelajari hubungan

kompleks antar variabel. Hasil akhir dari proses ini merupakan keluaran jaringan yang kemudian dibandingkan dengan target untuk menghitung tingkat kesalahan (*error*).

B. *Backpropagation*

Tahapan selanjutnya adalah umpan balik atau *backpropagation*, yaitu proses memperbaiki bobot dan bias berdasarkan nilai kesalahan yang diperoleh dari tahap forward propagation [22]. Proses ini menggunakan algoritma pembelajaran tertentu, misalnya gradient descent, untuk meminimalkan nilai error dengan menyesuaikan parameter jaringan secara bertahap. Proses *forward propagation* dan *backpropagation* dilakukan secara berulang-ulang (iteratif) hingga diperoleh bobot dan bias optimal yang menghasilkan kesalahan minimum pada keluaran jaringan. Dengan demikian, jaringan saraf tiruan dapat melakukan prediksi atau klasifikasi dengan tingkat akurasi yang tinggi.

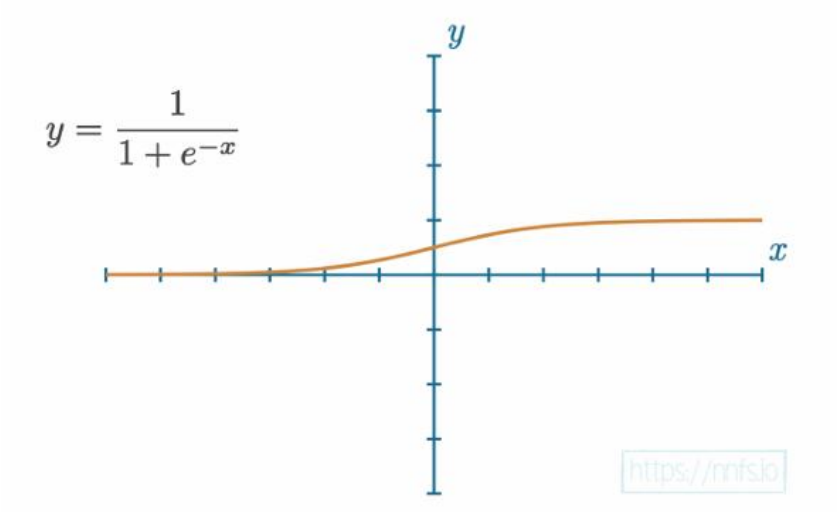
2.6.4 Jenis-jenis Fungsi Aktivasi

Fungsi aktivasi berfungsi sebagai penentu nilai output yang akan dikeluarkan suartu neuron dalam lebel aktivasi tertentu yang berguna untuk mengontrol nilai ouput agar sesuai sebelum diteruskan ke lapisan selanjutnya [22]. Berikut jenis-jenis dari fungsi aktivasi:

A. Fungsi Aktivasi Sigmoid

Fungsi aktivasi sigmoid merupakan fungsi aktibasi yang menghasilkan nilai kontinu antara 0–1 [26]. Ini memungkinkan model mengetahui seberapa kuat suatu neuron diaktifkan, sehingga proses pelatihan (*training*) dan optimisasi dapat berjalan lebih efektif.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

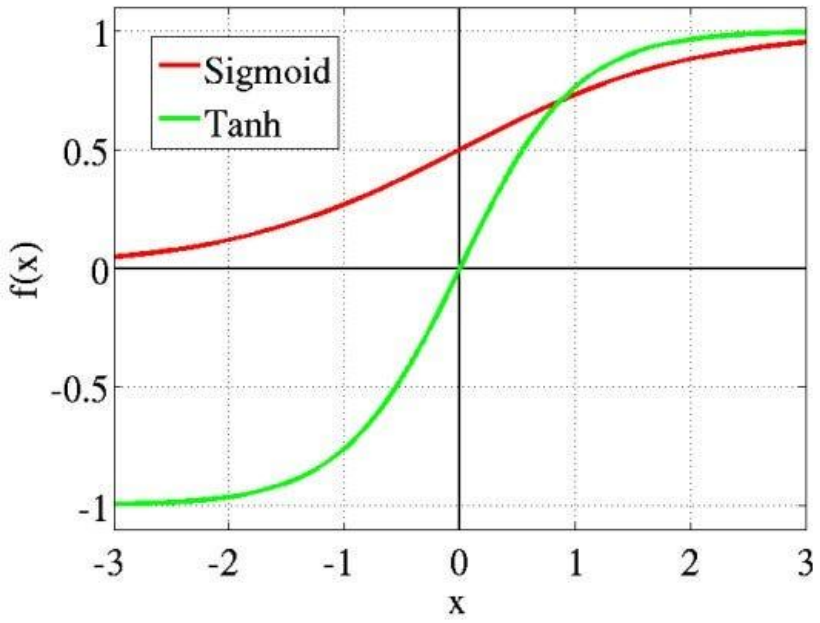


Gambar 10. Grafik Fungsi Aktivasi Sigmoid

B. Fungsi Aktivasi Tanh

Fungsi aktivasi Tanh merupakan fungsi aktivasi yang menghasilkan nilai output di rentang -1 – 1 yang dimana memiliki kelebihan data terpusat di sekitar nol, mempercepat konvergensi.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3}$$

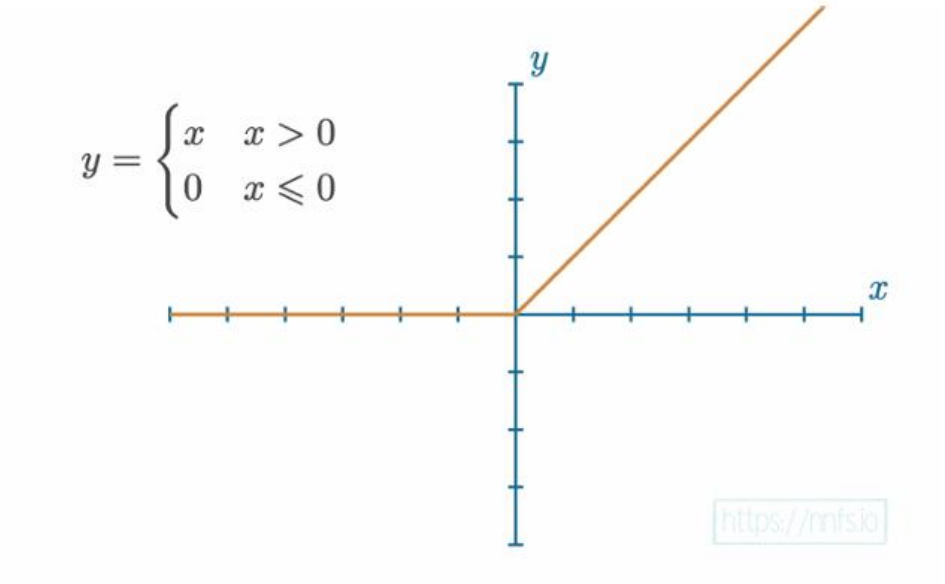


Gambar 11. Grafik Fungsi Aktivasi Tanh

C. Fungsi Aktivasi ReLU (*Rectified Linear Unit)*

Fungsi aktivasi ReLU merupakan fungsi aktivasi yang paling populer dan banyak digunakan pada jaringan saraf tiruan modern [22]. Fungsi aktivasi ini angat efisien secara komputasi karena tidak memerlukan perhitungan eksponensial seperti pada fungsi sigmoid atau tanh.

$$z = \sum_{i=1}^{n} x_i w_i + b \tag{4}$$

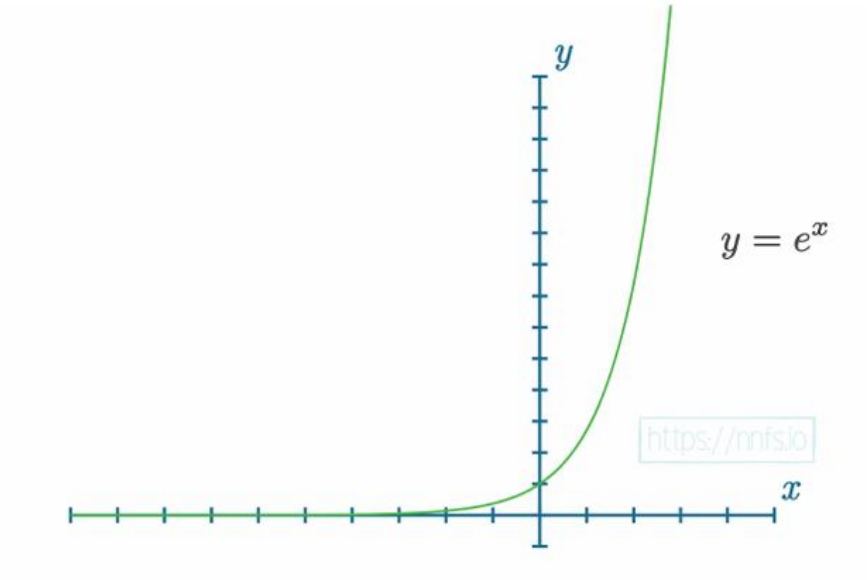


Gambar 12. Fungsi Aktivasi ReLU

D. Fungsi Aktivasi Softmax

Fungsi aktivasi softmax merupakan fungsi aktivasi yang menghasilkan output menjadi probabilitas antar kelas (0–1), dengan total 1. Fungsi ini biasanya digunakan untuk klasifikasi multi-kelas pada lapisan output.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{5}$$



Gambar 13. Fungsi Aktivasi Softmax

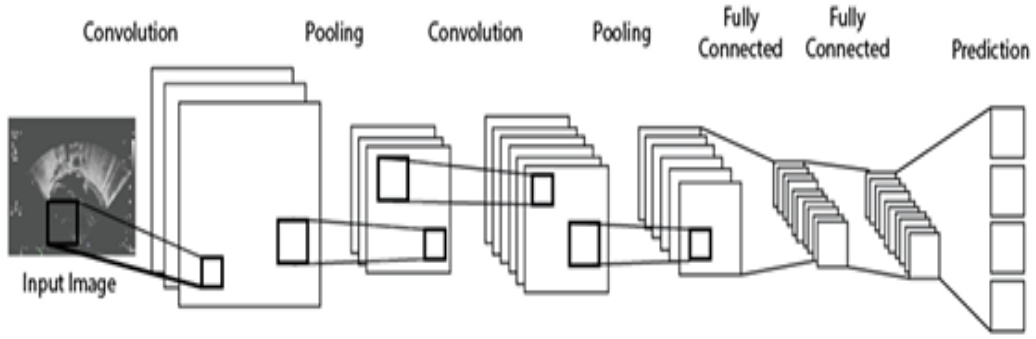
2.7 *Convolutional Neural Network*

Convolutional *Neural network* (CNN) merupakan salah satu arsitektur *Deep Learning* yang sangat populer dan banyak digunakan dalam bidang pengolahan citra (computer vision). CNN merupakan pengembangan dari *Multilayer Perceptron* (MLP) yang terinspirasi dari cara kerja jaringan saraf manusia dalam mengenali pola visual [24]. Jaringan ini terdiri dari sejumlah lapisan komputasi yang bekerja secara bertahap dalam mengekstraksi fitur dari data masukan, sehingga mampu menghasilkan representasi yang lebih bermakna untuk proses klasifikasi, deteksi, atau segmentasi gambar.

CNN pertama kali diperkenalkan oleh Yann LeCun dan rekan-rekannya pada tahun 1998 melalui model LeNet-5, yang berhasil diterapkan untuk pengenalan karakter tulisan tangan. Sejak saat itu, CNN berkembang pesat dan menjadi dasar dari berbagai model *Deep Learning* modern seperti AlexNet, VGGNet, ResNet, dan lainnya [21]. Dibandingkan dengan jaringan saraf tradisional, CNN memiliki keunggulan karena tidak semua neuron saling terhubung sepenuhnya; hanya neuron dalam area lokal tertentu yang berinteraksi. Hal ini memungkinkan CNN untuk mengenali pola spasial dengan lebih efisien dan mengurangi jumlah parameter yang harus dilatih.

2.7.1 Lapisan Utama CNN

Secara umum, arsitektur CNN terdiri atas tiga komponen utama, yaitu lapisan konvolusional (*convolutional layer*), lapisan pereduksi (*pooling layer*), dan lapisan terhubung penuh (*fully connected layer*) [24]. Lapisan konvolusional berfungsi mengekstraksi fitur penting dari citra melalui operasi konvolusi dengan *filter* atau kernel tertentu, sementara lapisan pooling bertujuan untuk mengurangi dimensi data tanpa kehilangan informasi penting guna meningkatkan efisiensi komputasi. Lapisan fully connected kemudian digunakan untuk mengintegrasikan hasil ekstraksi fitur dan melakukan proses klasifikasi terhadap data masukan.



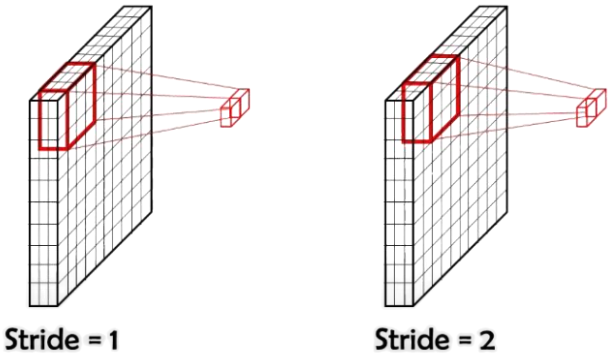
Gambar 14. Arsitektur CNN

A. *Convolutional layer* (Lapisan Konvolusi)

Lapisan convolutional (*convolutional layer*) merupakan bagian utama dari arsitektur Convolutional *Neural network* (CNN), di mana sebagian besar proses komputasi dilakukan. Pada lapisan ini, jaringan menggunakan *filter* atau kernel untuk mengekstraksi fitur penting dari citra [24]. *Filter* tersebut memiliki ukuran tertentu, misalnya 5 × 5 × 3, di mana angka 5 pertama menunjukkan tinggi, angka 5 kedua menunjukkan lebar, dan angka 3 mewakili jumlah kanal warna dari gambar (misalnya RGB). *Filter* ini kemudian digeser ke seluruh area gambar, dan pada setiap pergeseran dilakukan operasi dot product (perkalian titik) antara nilai-nilai piksel gambar dan nilai *filter* untuk menghasilkan activation map (*feature map*) yaitu hasil ekstraksi fitur dari proses konvolusi.

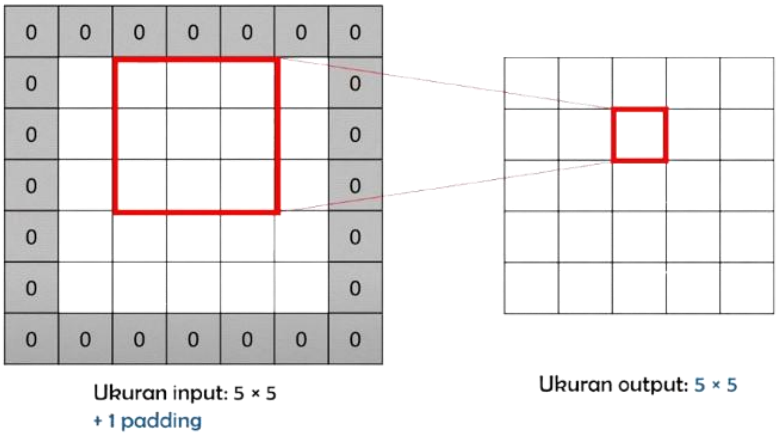
Dalam lapisan konvolusi ada beberapa parameter yang mempengaruhi hasil konvolusi yaitu sebagai berikut:

1. *Stride* merupakan parameter yang menentukan seberapa jauh *filter* bergeser pada setiap langkah, baik secara horizontal maupun vertical [24]. Jika nilai *stride* bernilai 1, maka *filter* bergerak satu piksel per langkah. Nilai *stride* yang kecil memungkinkan model menangkap detail yang lebih halus dari gambar, namun memerlukan waktu komputasi yang lebih besar. Sebaliknya, *stride* yang besar mempercepat proses, tetapi dapat menyebabkan hilangnya informasi detail.



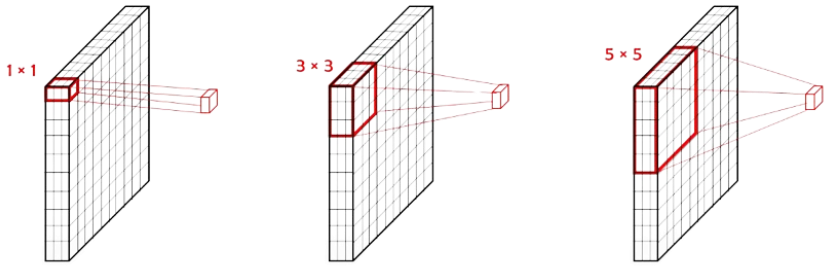
Gambar 15. *Stride*

2. *Padding,* atau sering disebut zero padding, adalah teknik menambahkan baris dan kolom piksel bernilai nol di tepi gambar sebelum dilakukan proses konvolusi [24]. Tujuannya adalah untuk mengontrol ukuran *feature map* hasil konvolusi agar tidak menyusut terlalu cepat serta menjaga agar informasi di tepi gambar tidak hilang.



Gambar 16. *Padding*

3. *Filter/kernel* adalah sebuah matriks kecil yang digunakan untuk mendeteksi pola seperti tepi, tekstur, atau bentuk tertentu dalam gambar.



Gambar 17. *Filter/Kernel*

Berikut ini rumus dasar perhitunagan pada lapisan konvolusi [27]:

$$O(i,j) = \sum_{M=0}^{M-1} \sum_{N=0}^{N-1} I(i+m, j+n) x\ K(m,n) \tag{6}$$

Keterangan :

1. *O(i , j)* = nilai piksel pada posisi (i, j) dari *feature map* (output)
2. *I (i+m , j+n)* = nilai piksel dari citra input
3. *K (m,n)* = nilai ddari *kernel/filter*
4. *M , N* = ukuran kernel (misalnya 3×3, 5×5, dll)

Berikut ini rumus perhitungan output jika pada lapisan konvolusi memiliki *stride* dan *padding* [27]:

$$O = \frac{(W - K\ 2P)}{S} + 1 \tag{7}$$

Keterangan :

1. *W* = ukuran input
2. *K* = ukuran kernel
3. *P* = *padding*
4. *S* = *stride*

B. *Pooling layer* (Lapisan Pooling)

Pooling layer merupakan lapisan setelah lapisan konvolusi yang memiliki fungsi untuk mengurangi jumlah parameter yang perlu dilatih sehingga ukuran data menjadi lebih kecil, efisien, dan mudah dikelola, sekaligus membantu mengontrol risiko overfitting [24]. Terdapat beberapa jenis teknik pooling yang umum digunakan, yaitu max pooling dan average pooling. Pada max pooling, nilai

tertinggi dalam area tertentu dari *feature map* akan dipilih sebagai hasilnya, sedangkan pada average pooling, nilai rata-rata dari area tersebut yang akan digunakan. Teknik ini membantu model mempertahankan fitur penting sekaligus mengurangi kompleksitas data.

Gambar 18. *Max Pooling*

Berikut ini rumus perhitungan pada *Max Pooling* pada lapisan *pooling :*

$$y_{(i,j)} = \frac{max}{(m,n)\in R(i,j)} x_{m,n} \tag{8}$$

Keterangan :

1. $y_{(i,j)}$ = nilai ouput pada posisi (I,j) setelah pooling
2. $x_{(m,n)}$ = nilai input pada posisi (m,n) dalam jendela pooling
3. $R(i,j)$ = area jendela pooling
4. *Max* = diambil nilai terbesar pada jendela tersebut

Gambar 19. *Average Pooling*

Berikut ini rumus perhitungan pada *Average pooling* pada lapisan pooling :

$$y_{(i,j)} = \frac{1}{kxk} \sum_{(m,n)\in R(i,j)} x_{m,n} \tag{9}$$

Keterangan :

1. k = ukuran kernel pooling

2. $R(i,j)$ = area pooling
3. $\Sigma x(m,n)$ = jumlah semua nilai di area tersebut

Kemudian untuk menghitung ukuran ouput setelah dilakukan pooling layer dapat dihitung dengan rumus berikut ini :

$$O = \frac{I - K}{S} + 1 \tag{10}$$

Keterangan :

1. O = ukuran ouput
2. I = ukuuran input
3. K = ukuran kernel pooling
4. S = *stride*

C. *Fully connected layer* (Lapisan Terhubung Penuh)

Fully connected layer (FC Layer) merupakan tahap akhir dalam arsitektur Convolutional *Neural network* (CNN) setelah proses feature extraction (lapisan konvolusi dan pooling). Pada tahap ini, *feature map* yang dihasilkan dari proses sebelumnya masih berbentuk array multidimensi, sehingga perlu dilakukan proses flattening atau reshape yaitu mengubah data dari bentuk matriks menjadi vektor satu dimensi [24]. Hasil vektor ini kemudian digunakan sebagai input ke dalam *Fully connected layer*, di mana setiap neuron pada layer ini terhubung secara penuh dengan neuron pada layer sebelumnya. FC Layer berfungsi untuk menggabungkan fitur-fitur yang telah diekstraksi agar dapat menghasilkan keputusan akhir, seperti klasifikasi atau prediksi. FC layer memiliki beberapa *hidden layer*, activation function, output layer dan *loss* function.

2.7.2 Arsitektur CNN

Sejak tahun 1989 hingga saat ini, CNN mengalami pengembangan dan perbaikan untuk menambah performa di berbagai kebutuhan. Perbaikan yang dilakukan meliputi pengoptimalan parameter, regulasi, kedalaman, dan lainnya yang membantu dalam menambah kinjerja CNN di berbagai bidang [21]. Berikut ini beberapa arsitektur CNN yang banyak digunakan yaitu sebagai berikut:

1. LeNet5, yang merupakan arsitektur awal CNN dengan memiliki 7 lapisan dengan 5 konvolusi , 2 lapisan pooling, 2 Lapisan Terhubung Penuh yang digunakan untuk pengenalan karakter tulisan tangan.
2. Alexnet yang meruapakan jaraingan Deep CNN pertama untuk meningkatkan generalisasi gambar dengan memiliki 8 lapisan dengan 5 konvolusi (max *pooling layer*s), dan 3 lapisan yang terhubung sepenuhnya.
3. VGGNet merupakan model CNN dengan arsitektur yang sederhana namun efektif. Ciri khasnya adalah penggunaan kernel konvolusi berukuran kecil, yaitu 3×3, yang disusun berlapis-lapis untuk mengekstraksi fitur secara mendalam. VGGNet memiliki 16 lapisan konvolusi yang diakhiri dengan dua lapisan fully connected sebagai bagian dari output layer.
4. GoogleNet merupakan model populer pertama yang memperkenalkan arsitektur jaringan yang lebih kompleks dengan beberapa cabang (multi-branch). GoogleNet dilatih menggunakan dataset ImageNet, yang mampu mengklasifikasikan citra ke dalam 1.000 kategori objek. Model ini terdiri dari 22 lapisan, dan arsitekturnya lebih efisien dibandingkan CNN tradisional dengan jumlah parameter yang lebih sedikit berkat desain modularnya.
5. ResNet50 merupakan varian dari arsitektur Residual *Network* (ResNet) yang memiliki 50 lapisan konvolusi dan 1 lapisan terhubung penuh. Ciri utama dari model ini adalah penggunaan shortcut connection (skip connection) untuk mengatasi masalah vanishing gradient, sehingga jaringan bisa dilatih jauh lebih dalam tanpa kehilangan akurasi.

2.8 *Computer Vision*

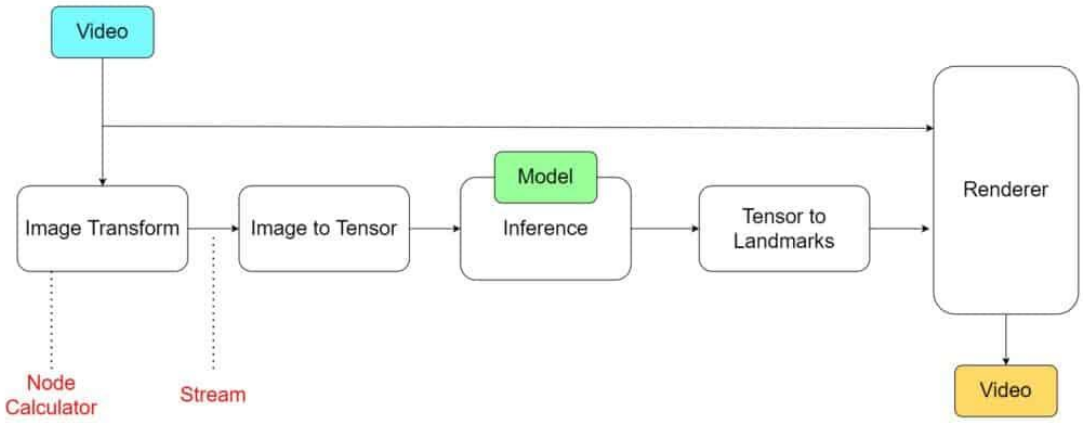
Computer Vision adalah bidang dalam kecerdasan buatan (*Artificial Intelligence*) yang berfokus pada bagaimana komputer dapat melihat, mengenali, dan memahami gambar atau video seperti halnya manusia menggunakan penglihatannya [27], [28]. *Computer Vision* digunakan untuk dapat membuat system atau computer mengenali suatu objek, maupun pola, visual seperti gambar atau video, lalu mengambil insight dari data tersebut. Saat ini, *Computer Vision* sudah banyak digunakan dalam beberapa kebutuhan yaitu OCR (*Optical Character Recognition*) yang dapat digunakan untuk membaca tulisan tangan, inspeksi industri seperti untuk

mendeteksi cacat pada komponen manufaktur industry, medis pencocokan citra operasi dan analisis otak, dan banyak lainnya.

2.9 MediaPipe

MediaPipe merupakan open-source *frame*work yang dikembangkan oleh Google untuk membangun *Machine Learning* pipeline lintas platform yang mampu memproses berbagai jenis data stream seperti video, audio, maupun time-series data secara *Real-time*[11] . *Frame*work ini pertama kali diperkenalkan untuk analisis video dan audio di YouTube, dan sejak dirilis secara publik pada tahun 2019 [10].

MediaPipe terdiri atas dua komponen utama, yaitu MediaPipe *Frame*work dan MediaPipe Solutions [10]. MediaPipe *Frame*work dikembangkan menggunakan bahasa pemrograman C++, Java, dan Objective-C, serta memiliki tiga API penting, yaitu Calculator API, Graph Construction API, dan Graph Execution API. Komponen ini berfungsi untuk mendefinisikan dan menjalankan alur pemrosesan data melalui struktur graf yang menghubungkan berbagai node atau calculator untuk melakukan tugas-tugas tertentu. Sementara itu, MediaPipe Solutions mencakup serangkaian model pra-latih berbasis TensorFlow dan TensorFlow Lite yang dibangun untuk kasus penggunaan spesifik seperti face detection, pose estimation, dan hand tracking.



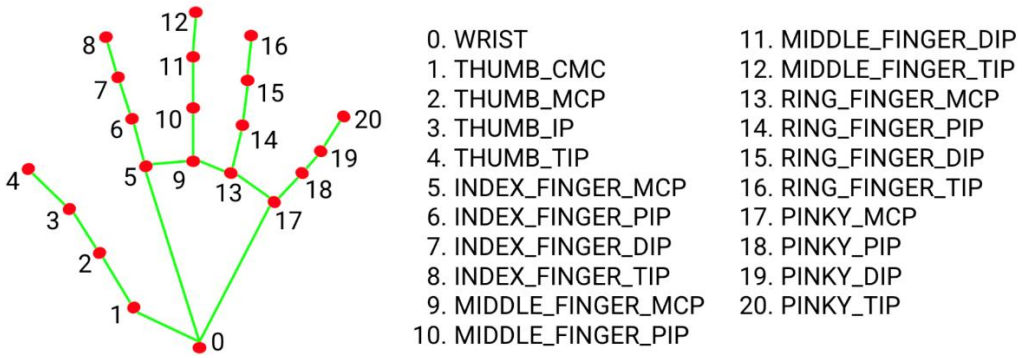
Gambar 20. Bagan Alir Cara Kerja MediaPipe [29]

Berikut ini penjelasan dari bagan alir pada Gambar 20:

1. Video Input: video yang ditangkap melalui kamera atau file video yang dimasukkan ke dalam sistem. Setiap *frame* video yang diterima akan menjadi dataset yang akan diproses oleh pipeline MediaPipe.
2. *Image Transform*: MediaPipe melakukan proses pra-pemrosesan terhadap setiap *frame* yang masuk. Proses ini meliputi pengubahan ukuran (*resize*), normalisasi warna, penyesuaian orientasi gambar, serta konversi format yang diperlukan.
3. *Image to Tensor: Frame* yang telah diproses kemudian diubah menjadi tensor, yaitu representasi numerik yang digunakan oleh model deep learning. Proses ini mengonversi piksel gambar menjadi matriks numerik, menambahkan dimensi batch, serta menyesuaikan bentuk tensor agar kompatibel dengan model.
4. *Model Inference:* Model MediaPipe melakukan inferensi terhadap tensor yang diberikan untuk mendeteksi objek atau landmark seperti tangan, wajah, atau pose tubuh tergantung tugas yang digunakan. Pada tahap ini model CNN akan memprediksi lokasi titik-titik penting, menghasilkan *Bounding Box* jika diperlukan, serta mengeluarkan nilai kepercayaan (confidence).
5. *Tensor to Landmarks:* Output tensor dari model kemudian diterjemahkan menjadi koordinat landmark yang lebih mudah dipahami. MediaPipe melakukan post-processing berupa normalisasi posisi titik, konversi koordinat ke ruang gambar asli, dan penerapan smoothing untuk mengurangi noise antar *frame* sehingga landmark lebih stabil. Hasilnya adalah koordinat landmark lengkap seperti 21 titik tangan, 33 titik pose, atau 468 titik wajah.
6. *Renderer:* Tahap ini bertugas menampilkan hasil deteksi ke dalam bentuk visual. Renderer menggambar titik-titik landmark, garis penghubung, atau *Bounding Box* pada *frame* video.
7. *Video Output:* Tahap terakhir adalah keluaran video yang telah dilengkapi dengan visualisasi landmark. Video ini dapat ditampilkan secara *real-time* atau disimpan menjadi file.

Salah satu solution yang paling populer dari MediaPipe adalah *Hand Tracking*, yang terdiri dari dua model utama, yaitu *Palm Detection* Model dan *Hand*

Landmark Model [10]. Model pertama berfungsi mendeteksi area telapak tangan dalam gambar, karena bagian tersebut lebih kaku dan mudah dikenali dibandingkan keseluruhan tangan. Setelah area telapak terdeteksi, citra tersebut dipotong dan diteruskan ke Hand *Landmark* Model, yang kemudian mengekstraksi 21 titik koordinat tiga dimensi (3D keypoints) pada tangan. Model ini dilatih menggunakan sekitar 30.000 citra dunia nyata yang telah diberi anotasi secara manual, sehingga mampu mendeteksi titik-titik tangan dengan presisi tinggi bahkan ketika tangan tidak sepenuhnya terlihat.



Gambar 21. *Landmark* Tangan Menggunakan Mediapipe

MediaPipe tidak memerlukan perangkat keras dengan spesifikasi tinggi karena dapat berjalan efisien baik pada CPU maupun GPU tanpa membutuhkan daya pemrosesan tambahan [30]. Dataset yang dihasilkan oleh MediaPipe biasanya berisi koordinat 21 titik tangan untuk setiap gambar atau *frame*, dan telah digunakan untuk berbagai aplikasi. Dengan kemampuannya dalam mendeteksi dan melacak titik-titik tangan secara akurat dan *Real-time*,

2.10 *You Only Look Once* (YOLO)

You Only Look Once (YOLO) merupakan salah satu algoritma object detection berbasis Convolutional *Neural network* (CNN) yang dirancang untuk mendeteksi dan mengenali objek secara cepat dan akurat . YOLO pertama kali diperkenalkan oleh Joseph Redmon pada tahun 2015 dengan tujuan utama melakukan deteksi objek secara *Real-time*. Berbeda dengan metode deteksi objek dua tahap seperti R-CNN atau Faster R-CNN yang memisahkan proses region proposal dan klasifikasi, YOLO bekerja dengan pendekatan satu tahap (*single-stage detector*), di mana proses deteksi dan klasifikasi dilakukan dalam satu jaringan tunggal [14] .

YOLO dalam deteksi objek menggunakan *Bounding Box* dan probabilitas kelas dari seluruh gambar sekaligus. Citra input dibagi menjadi grid berukuran S×S, kemudian pada setiap grid dihitung *Bounding Box* yang berisi objek serta nilai *confidence score* yang menunjukkan tingkat keyakinan model terhadap keberadaan objek tersebut. Pendekatan ini membuat YOLO memiliki kecepatan deteksi yang jauh lebih tinggi dibandingkan metode lain, menjadikannya sangat ideal untuk aplikasi *Real-time*.

Untuk mengukur kinerja deteksi objek, YOLO umumnya dievaluasi menggunakan dataset standar seperti Microsoft COCO (*Common Objects in Context*) dan PASCAL VOC (*Visual Object Classes*). Dataset COCO berisi 328.000 gambar dengan lebih dari 2,5 juta label dari 91 kategori objek, sedangkan PASCAL VOC berisi 11.530 gambar dari 20 kategori. Evaluasi dilakukan menggunakan metrik seperti *Precision*, *Recall*, *mean Average Precision* (mAP), serta *confusion matrix* yang menampilkan jumlah prediksi benar dan salah untuk setiap kelas objek.

YOLO telah digunakan di berbagai bidang. Pada bidang transportasi dan keamanan, YOLO dimanfaatkan dalam sistem kendaraan otonom, pengenalan plat nomor, deteksi pejalan kaki, sistem parkir, serta deteksi kapal dan pesawat di bidang pertahanan. Di bidang medis, YOLO berperan penting dalam deteksi sel kanker, polip, patah tulang, dan membantu tenaga medis dalam diagnosis penyakit kulit. YOLO juga telah diterapkan dalam pengenalan bahasa isyarat, di mana algoritma ini digunakan untuk mendeteksi posisi dan gerakan tangan secara akurat untuk mendukung sistem penerjemah bahasa isyarat otomatis.

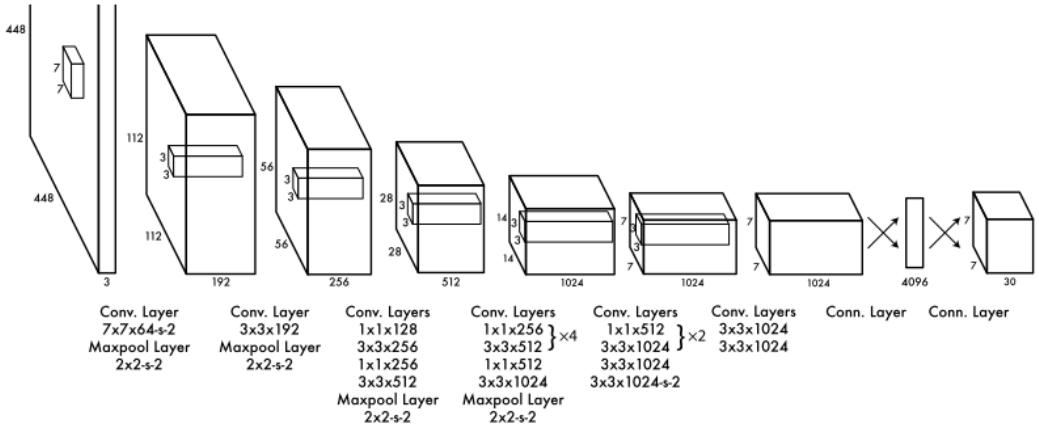
2.10.1 Perkembangan Versi YOLO

Algoritma YOLO (*You Only Look Once*) pertama kali dikembangkan oleh Joseph Redmon pada tahun 2015 sebagai solusi *Real-time* object detection yang cepat dan efisien. Seiring waktu, YOLO telah mengalami banyak penyempurnaan dari versi pertama hingga versi ke-11, dengan setiap versi membawa inovasi baru baik dari sisi arsitektur, kecepatan, maupun akurasi.

Gambar 22. Perkembangan YOLO

A. YOLOv1

YOLOv1 Merupakan versi awal yang memandang deteksi objek sebagai permasalahan regresi tunggal. Gambar dibagi menjadi grid (S×S), dan setiap grid memprediksi *Bounding Box* beserta *confidence score*. Arsitekturnya terdiri dari 24 lapisan konvolusi dan 2 *lapisan fully connected*, dengan dasar GoogLeNet [14]. Versi 1 cukup cepat tetapi, kurang akurat dalam mendeteksi objek kecil dan objek yang berdekatan.



Gambar 23. Arsitektur YOLOv1

Arsitektur YOLOv1 dimulai dari input gambar masukan berukuran 448×448×3. Proses dimulai dengan lapisan konvolusi 7×7×64 dengan *stride* 2 untuk menangkap fitur global, diikuti oleh max pooling 2×2 guna mengurangi ukuran spasial. Kemudian jaringan melanjutkan dengan lapisan 3×3×192 untuk memperdalam ekstraksi fitur dan kembali dilakukan pooling untuk menurunkan resolusi menjadi 56×56×192. Setelah itu, YOLOv1 menggunakan beberapa blok konvolusi berulang seperti 1×1×128 → 3×3×256 → 1×1×256 → 3×3×512, di mana lapisan 1×1 berfungsi mengurangi jumlah kanal fitur dan 3×3 berfungsi menangkap pola yang lebih kompleks. Proses ini terus berlanjut dengan jumlah *filter* yang meningkat hingga mencapai 1024 untuk memperkaya representasi fitur dari gambar.

Setelah seluruh lapisan konvolusi selesai, dihasilkan *feature map* berukuran 7×7×1024, yang kemudian diratakan dan diteruskan ke dua fully connected layer. Lapisan pertama memiliki 4096 neuron, sedangkan lapisan kedua menghasilkan output akhir berupa prediksi untuk setiap sel grid berukuran 7×7. Setiap sel memprediksi beberapa bounding *box*, nilai confidence, dan probabilitas kelas objek. Melalui kombinasi seluruh lapisan ini, YOLOv1 dapat melakukan deteksi objek secara menyeluruh hanya dalam satu proses inferensi, menjadikannya model deteksi *real-time* pertama yang cepat sekaligus cukup akurat.

B. YOLOv2

YOLOv2 menggunakan arsitektur baru bernama DarkNet-19 serta *Batch Normalization dan Anchor Boxes*. Model ini mampu mendeteksi hingga 9000 kelas karena pelatihan gabungan antara dataset ImageNet dan COCO. YOLOv2 juga menggunakan resolusi input dinamis (320–608 piksel) dan memperbaiki deteksi objek kecil melalui *feature map* berukuran 26×26 [14]. Keunggulan yaitu peningkatan akurasi dan kemampuan mendeteksi objek kecil dibandingkan versi sebelumnya. Tetapi model ini masih sering mengalami kesalahan dalam menentukan posisi objek (*localization error*) dan membutuhkan daya komputasi yang lebih besar dibanding YOLOv1.

C. YOLOv3

YOLOv3 merupakan versi dengan arsitektur yang lebih dalam menggunakan DarkNet-53 dan penambahan Residual Connection serta *Feature Pyramid Network* (FPN). Model ini memungkinkan deteksi multi-skala dengan tiga ukuran peta fitur (13×13, 26×26, dan 52×52) untuk meningkatkan performa terhadap objek berukuran kecil maupun besar [14]. YOLOv3 sangat bagus dalam kestabilan deteksi pada berbagai ukuran objek, tapi memiliki ukuran model yang cukup besar dan memerlukan perangkat keras GPU dengan kapasitas tinggi untuk pelatihannya.

D. YOLOv4

YOLOv4 melakukan perkembangan dengan penggunaan CSPDarknet53 sebagai *backbone*, serta *Spatial Pyramid Pooling (SPP)* dan *Path Aggregation Network* (PANet) pada bagian *neck*. OLOv4 juga menggunakan *Bag of Freebies* (BoF) dan *Bag of Specials* (BoS) yang meningkatkan akurasi tanpa mengurangi kecepatan

inferensi. Aktivasi Mish, DropBlock Regularization, dan Mosaic Data Augmentation juga memperkuat kemampuan generalisasi model [14]. YOLOv4 unggul karena seimbang antara akurasi dan kecepatan, serta dapat dijalankan pada GPU kelas menengah. Tetapi kompleksitas arsitekturnya meningkat, sehingga pelatihan membutuhkan waktu lebih lama.

E. YOLOv5

YOLOv5 dikembangkan oleh Ultralytics dengan basis *frame*work PyTorch. Model ini membagi menjadi lima varian berbeda, yaitu YOLOv5n, s, m, l, dan x, yang dapat disesuaikan dengan kebutuhan perangkat. Struktur arsitekturnya menggunakan CSPDarknet53 sebagai *backbone*, SPPF (*Spatial Pyramid Pooling-Fast*), serta CSP-PANet sebagai *neck*, dengan fungsi aktivasi SiLU. YOLOv5 lebih ringan, modular, dan mudah digunakan, serta memiliki pelatihan yang lebih stabil berkat AutoAnchor dan *Exponential Moving Average* (EMA) [14].

F. YOLOv6

YOLOv6 dikembangkan oleh Meituan dan berfokus pada efisiensi industri. Model ini memperkenalkan EfficientRep *Backbone* berbasis RepVGG dan CSPStackRep, serta *anchor-free head* yang mengurangi beban komputasi [14]. Dan juga *Task Alignment Learning* (TAL) dan VariFocal *Loss* digunakan untuk meningkatkan kualitas prediksi. YOLOv6 memiliki keunggulan dari sisi efisiensi dan kecepatan inferensi, tetapi masih sedikit kalah dalam akurasi dibandingkan YOLOv7 pada dataset besar.

G. YOLOv7

YOLOv7 dikembangan oleh Wang dan tim dengan memperkenalkan arsitektur E-ELAN (*Extended Efficient Layer Aggregation Network*) yang meningkatkan kemampuan pembelajaran fitur tanpa kehilangan stabilitas [14]. YOLOv7 juga menambahkan *trainable bag-of-freebies, planned re-parameterization convolution* (RepConvN), serta *decoupled head* dengan *auxiliary head* untuk mempercepat konvergensi model. Versi ini menjadi salah satu yang paling seimbang antara kecepatan dan akurasi, tetapi kompleksitas pelatihannya relatif tinggi.

H. YOLOv8

YOLOv8 dikembangkan oleh Ultralytics dengan desain yang lebih sederhana namun sangat efisien[14] . Model ini menggantikan blok CSP dengan C2f module, menggunakan struktur *anchor-free* dan *decoupled head*, serta mendukung berbagai tugas seperti deteksi objek, segmentasi, klasifikasi, dan pose estimation. YOLOv8 sangat ringan, mudah digunakan, dan kompatibel dengan berbagai perangkat.

I. YOLOv9

YOLOv9 memiliki 2 pengembangan, yaitu Programmable Gradient Information (PGI) dan GELAN (*Generalized Efficient Layer Aggregation Network*) [14]. PGI memungkinkan pelatihan jaringan dalam yang lebih stabil dengan mempertahankan informasi gradien, sedangkan GELAN meningkatkan efisiensi tanpa mengorbankan akurasi. YOLOv9 unggul karena mampu mencapai akurasi tinggi dengan jumlah parameter lebih sedikit, namun arsitekturnya lebih kompleks dibanding pendahulunya.

J. YOLOv10

YOLOv10 dikembangkan oleh Tsinghua University dengan pendekatan baru yang menghapus proses *Non-Maximum Suppression* (NMS) dan menggantinya dengan consistent binary assignment untuk menyederhanakan tahap akhir inferensi. Selain itu, digunakan *Partial Self-Attention (*PSA) dan Compact Inverted Block (CIB) untuk mengurangi beban komputasi [14]. YOLOv10 lebih efisien dan cepat, tetapi memerlukan penyetelan parameter yang lebih sensitif agar performanya optimal.

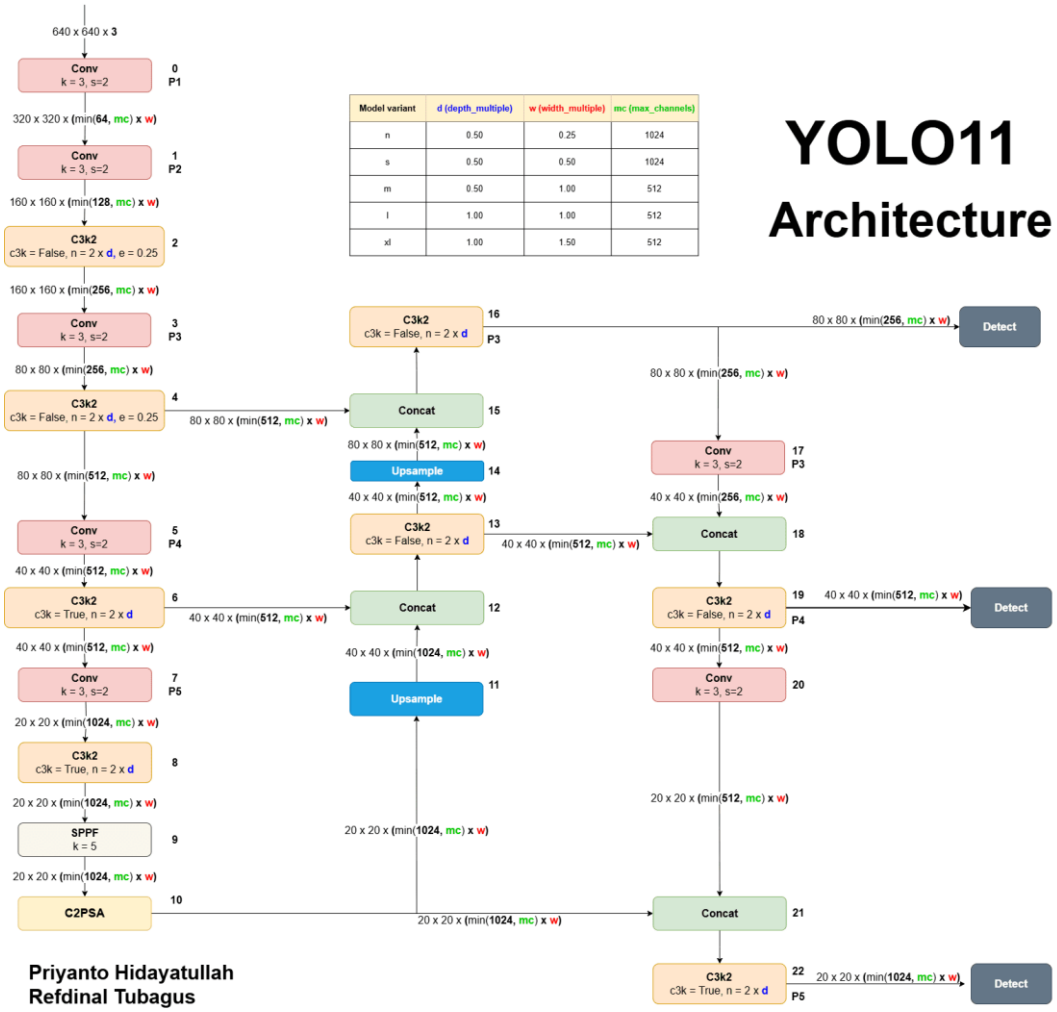
K. YOLOv11

YOLOv11 merupakan versi terbaru yang dikembangkan oleh Ultralytics dengan basis arsitektur YOLOv8 tetapi ditambahkan dua modul baru, yaitu C3k2 dan C2PSA. C3k2 digunakan untuk meningkatkan efisiensi komputasi pada *backbone*, sedangkan C2PSA mengintegrasikan *Partial Self-Attention* untuk meningkatkan fokus model pada area penting gambar [14]. YOLOv11 juga mendukung *Oriented Bounding Boxes* (OBB), yang memungkinkan deteksi objek dengan orientasi miring, sangat berguna untuk citra satelit dan kendaraan otonom. Versi ini memiliki kecepatan tinggi, efisiensi energi lebih baik, serta akurasi lebih tinggi dibanding

YOLOv8. Karena merupakan versi terbaru, dokumentasi dan optimisasi lanjutan masih terus dikembangkan oleh komunitas.

2.10.2 Arsitektur YOLOv11

Versi YOLOv11 merupakan evolusi terkini dari seri algoritma deteksi objek YOLO yang dirancang untuk meningkatkan performa deteksi objek secara *Real-time* dan memperluas fungsionalitas ke berbagai tugas visi komputer seperti segmentasi, klasifikasi, serta deteksi orientasi (*Oriented Bounding Boxes* – OBB).



Gambar 24. Arsitektur YOLOv11

Arsitektur YOLOv11 dibangun dalam tiga komponen utama: *Backbone*, *Neck*, dan Head. *Backbone* (bagian kuning) berfungsi untuk mengekstraksi fitur awal dari gambar masukan (640 x 640 x 3). Pada bagian ini menggunakan serangkaian blok Conv (Konvolusi), blok c2f (modul *Cross-Stage Partial* yang disempurnakan), dan

lapisan SCDown (*Spatial Channel Downsampling*) untuk secara progresif mengurangi dimensi spasial (*width* dan *height*) dengan meningkatkan jumlah saluran (*channels*), menghasilkan peta fitur (*feature maps*) dengan banyak konteks pada skala yang berbeda. Bagian akhir *Backbone* adalah modul SPPF (*Spatial Pyramid Pooling Fast*) untuk agregasi fitur pada berbagai skala, diikuti oleh lapisan PSA (*Parallel Spatial Attention*), yang merupakan pengembangan dalam V11 untuk meningkatkan pemrosesan informasi spasial sebelum diteruskan ke *Neck*.

Neck (bagian hijau) bertugas mengagregasi fitur-fitur yang diekstraksi dari *Backbone* pada berbagai skala. Bagian ini menggunakan kombinasi operasi Upsample untuk meningkatkan resolusi fitur skala dalam (*deep*) dan lapisan Concat untuk menggabungkannya dengan fitur skala dangkal (shallow) dari *Backbone*. Proses ini melibatkan blok C2fCIB (*blok Cross-Stage Partial* dengan C1B yang dimodifikasi, yang merupakan modul bottle*neck* internal menggunakan konvolusi kedalaman terpisah 3x3DW), memastikan informasi spasial dan semantik dipertahankan dan diperkaya. *Neck* menjalankan proses *Feature Pyramid Network* (FPN) dan *Path Aggregation Network* (PAN) yang dimodifikasi untuk menghasilkan *feature maps* yang siap untuk prediksi pada skala output yang berbeda.

Head (bagian merah muda) adalah bagian yang mengambil *feature maps* yang disempurnakan dari *Neck* dan melakukan prediksi akhir. YOLOv11 menggunakan *Head* yang terbagi menjadi lapisan *one-to-one head* dan *one-to-many head*. Lapisan *one-to-one head* umumnya digunakan untuk prediksi *anchor-free* utama, sedangkan *one-to-many head* dapat digunakan untuk refinement atau tugas tambahan lainnya, memungkinkan model untuk mencapai akurasi tinggi sambil mempertahankan kecepatan inferensi yang baik (desain *decoupled head*). Struktur ini memungkinkan prediksi yang akurat dari *Bounding Box* dan klasifikasi kelas secara *Real-time*.

YOLOv11 memiliki beberapa versi sehingga pengguna dapat menggunakan versi model sesuai kebutuhan dan spesifikasi perangkat [31]. Berikut versi dari YOLOv11 :

Tabel 1. Versi dari YOLOv11

Versi	Param(juta)	FLOPs (G)	mAP@0.5-0.95	Kecepatan GPU T4 (ms)
YOLOv11n (Nano)	2.6	6.5	39.5	1.5
YOLOv11s (Small)	9.4	21.5	47.0	2.5
YOLOv11m (Medium)	20.1	68.0	51.5	4.7
YOLOv11l (Large)	25.3	86.9	53.4	6.2
YOLOv11x (Extra Large)	56.9	194.9	54.7	11.3

2.10.3 Hyperparameter YOLOv11

Dalam pelatihan model YOLOv11 terdapat beberapa hyperparameter penting yang dapat mempengaruhi performa dari model tersebut seperti *batch size, epoch, learning rate*, dan *optimizer*. Berikut penjelasannya :

1. *Batch size* adalah jumlah data yang diproses dalam satu iterasi pelatihan sebelum model memperbarui bobotnya [32]. Ukuran batch yang kecil (misalnya 8 atau 16) membuat model lebih sering memperbarui bobot sehingga dapat beradaptasi lebih cepat, tetapi cenderung menghasilkan fluktuasi pada nilai *loss*. Sebaliknya, batch size yang besar (misalnya 64 atau 128) memberikan estimasi gradien yang lebih stabil dan efisien untuk pemrosesan GPU, tetapi membutuhkan memori lebih besar dan kadang menyebabkan model sulit mencapai konvergensi optimal.
2. Epoch menunjukkan berapa kali seluruh dataset dilalui dalam proses pelatihan. Semakin besar jumlah epoch, semakin lama model belajar dari data, tetapi risiko

overfitting juga meningkat jika model terlalu lama dilatih pada data yang sama. Biasanya, pemilihan jumlah epoch dilakukan berdasarkan hasil pengamatan terhadap nilai *loss* dan akurasi pada data validasi.

3. *Optimizer* merupakan algoritma yang digunakan untuk memperbarui bobot model agar *loss* function dapat diminimalkan. Beberapa optimizer populer antara lain Stochastic Gradient Descent (SGD), Adam, RMSProp, dan Adagrad [33]. Adam adalah yang paling sering digunakan karena mengombinasikan keunggulan Momentum dan Adaptive Learning Rate, sehingga konvergensi lebih cepat dan stabil.
4. *Learning rate* adalah besarnya langkah pembaruan bobot dalam setiap iterasi. Nilai learning rate yang terlalu besar dapat menyebabkan model gagal konvergen karena melompat terlalu jauh dari titik minimum, sedangkan nilai yang terlalu kecil membuat proses pelatihan menjadi lambat dan berpotensi terjebak di local minima.

2.11 Tahapan Penelitian

Penelitian ini dilakukan melalui beberapa tahapan yaitu meliputi studi literatur, pengumpulan data, *preprocessing* data, pelatihan data, evaluasi model, dan pengujian *realtime* model yang dimana tahapan ini memiliki referensi dari penelitian sebelumnya [34], [35], [36]. Berikut ini penjelasan masing masing tahapan:

2.11.1 Kajian Pustaka dan Lapangan

Tahapan kajian pustaka dan lapangan dilakukan sebagai langkah awal untuk memperoleh pemahaman mendalam mengenai topik penelitian. Tahap ini meliputi studi literatur terhadap berbagai referensi seperti jurnal ilmiah, buku, artikel, serta penelitian terdahulu yang relevan dengan sistem penerjemah bahasa isyarat, metode deteksi objek menggunakan YOLO, dan pemanfaatan MediaPipe. Selain itu, dilakukan pula observasi dan wawancara langsung dengan pihak terkait untuk mengetahui kondisi nyata di lapangan serta kebutuhan pengguna. Tujuan dari tahapan ini adalah memperkuat landasan teori, memahami perkembangan teknologi terkini, serta mengidentifikasi kekurangan penelitian sebelumnya maupun

kebutuhan lapangan yang dapat dijadikan dasar dalam perancangan sistem yang lebih baik.

2.11.2 Pengumpulan Data

Tahapan pengumpulan data dilakukan untuk memperoleh dataset yang akan digunakan dalam pelatihan dan pengujian model. Tujuan dari tahap ini adalah memastikan ketersediaan data yang relevan, representatif, dan berkualitas untuk digunakan dalam pelatihan dan pengujian model..

2.11.3 *Preprocessing* Data

Tahap *preprocessing* data bertujuan untuk menyiapkan dan meningkatkan kualitas data sebelum digunakan dalam pelatihan model. Beberapa langkah yang dilakukan meliputi

1. Pembagian dataset, yaitu data juga dibagi menjadi *training* set, *validation* set, dan *testing* set untuk memastikan proses pelatihan dan evaluasi berjalan optimal.
2. *Augmentasi data*, yaitu memperbanyak variasi dataset untuk menghindari *overfitting* dan meningkatkan kemampuan generalisasi model. Augmentasi data citra terbagi menjadi tiga kategori utama untuk meningkatkan variasi dataset pelatihan: Transformasi Geometris berfokus pada perubahan posisi dan orientasi objek dalam citra seperti Pembalikan (Flipping) horizontal/vertikal, Rotasi (Rotation) pada sudut tertentu, Penskalaan (Scaling) ukuran gambar, Pemotongan (Cropping) acak untuk fokus pada sebagian citra, Translasi (Translation) atau penggeseran posisi, dan Pergeseran Shear (Shearing) untuk memiringkan perspektif gambar. Selanjutnya, Transformasi Warna dan Intensitas memanipulasi nilai pixel untuk mensimulasikan kondisi pencahayaan yang berbeda, meliputi Perubahan Kecerahan (Brightness) dan Kontras (Contrast), penyesuaian Saturasi dan Hue, serta Penambahan Noise (seperti Gaussian Noise) dan Pengubahan Ruang Warna (misalnya, RGB ke HSV). Terakhir, Teknik Augmentasi Lanjutan mencakup CutOut untuk menghapus area kecil secara acak, Mixup yang mencampur pixel dan label dari dua gambar berbeda secara li*near*, dan AugMix yang menggabungkan berbagai

teknik augmentasi untuk melatih model yang sangat tahan terhadap corruptions.

3. Selanjutnya dilakukan *ekstraksi fitur* untuk mendapatkan informasi penting dari gambar yang akan digunakan pada proses pelatihan. Ekstraksi fitur membantu model dalam mengenali pola atau karakteristik visual yang menjadi pembeda antara masing-masing gestur bahasa isyarat.
4. Proses *resize* yaitu mengubah ukuran gambar dataset agar semua gambar memiliki ukuran yang seragam sesuai kebutuhan model.
5. Proses *normalisasi* dilakukan dengan menyesuaikan nilai piksel gambar ke rentang tertentu, yaitu 0 hingga 1. Normalisasi ini bertujuan untuk menstabilkan proses perhitungan pada model dan meningkatkan efisiensi selama proses pelatihan, sehingga model dapat belajar dengan lebih optimal. Normalisasi yang biasa digunakan yaitu normalisasi Min-Max yang digunakan untuk mengubah nilai piksel dari rentang 0–255 menjadi 0–1. Berikut rumus normalisasi Min-Max :

$$x` = \frac{x - x_{min}}{x_{\max -} x_{min}} \tag{11}$$

2.11.4 Pelatihan Model

Pada tahap pelatihan model, dataset yang telah diproses dimasukkan ke dalam arsitektur model yang digunakan, seperti YOLOv11. Model akan belajar mengenali pola dan fitur dari data menggunakan parameter dan *hyperparameter* seperti *batch size*, *learning rate*, *optimizer*, dan jumlah *epoch*. Tujuan dari tahap ini adalah untuk menghasilkan model terbaik yang memiliki tingkat akurasi tinggi dalam mendeteksi dan mengenali isyarat tangan. Selama proses pelatihan, dilakukan pemantauan terhadap nilai *loss* dan *accuracy* untuk memastikan model mengalami konvergensi dengan baik dan tidak terjadi *overfitting*.

2.11.5 Evaluasi Model

Evaluasi model merupakan tahap penting dalam pengembangan sistem berbasis *Machine Learning* dan *Deep Learning*, karena bertujuan untuk mengukur sejauh mana model mampu mengenali pola dan menghasilkan prediksi yang akurat

terhadap data baru. Pada penelitian ini yang merupakan mengevaluasi performa model klasifikasi bahasa isyarat yag bersifat Isolated SLR maka umumnya menggunakan metode *Accuracy*, *Precision*, *Recall*, and *F1-Score* [20]. Tetapi saat melakukan evaluasi performa menggunakan model YOLOv11 tidak dapat dilakukan hanya dengan metrik akurasi tersebut karena YOLOv11 tidak hanya melakukan klasifikasi terhadap objek, tetapi juga mendeteksi posisi objek di dalam citra melalui koordinat bounding *box*. Sehingga metrik evaluasi yang digunakan harus dapat menilai kedua hal tersebut yaitu ketepatan dalam mengenali kelas objek dan ketepatan dalam menentukan posisi objek.

Beberapa metrik utama yang digunakan untuk mengevaluasi kinerja model YOLOv11 antara lain *Precision*, *Recall*, *F1-Score*, *Intersection over Union* (IoU), dan *mean Average Precision* (mAP) [33]. Berikut ini penjelasan dari metrik tersebut:

1. *Precision*

Precision mengukur seberapa banyak prediksi positif yang benar dari seluruh prediksi positif yang dihasilkan model. Nilai *Precision* yang tinggi menunjukkan bahwa model jarang memberikan deteksi yang salah (*false positive*).

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

TP menunjukkan *True Positive* yang berarti deteksi benar sesuai label yang ada, dan FP menunjukkan *False Positive* yang berarti deteksi salah terhadap objek yang tidak ada.

2. *Recall*

Recall mengukur seberapa banyak objek sebenarnya yang berhasil dideteksi oleh model dari seluruh objek yang seharusnya terdeteksi. Nilai *Recall* yang tinggi menunjukkan bahwa model mampu mendeteksi sebagian besar objek tanpa banyak terlewat (*false negative*).

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

FN (*False Negative)* berarti Objek yang seharusnya terdeteksi, tetapi tidak terdeteksi model.

3. *F1-Score*

F1-Score merupakan metrik gabungan antara *Precision* dan *Recall* yang memberikan ukuran keseimbangan antara keduanya, terutama ketika distribusi data tidak seimbang.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{14}$$

Nilai *F1-Score* yang tinggi menunjukkan bahwa model tidak hanya akurat, tetapi juga konsisten dalam mendeteksi objek tanpa banyak kesalahan.

4. *Intersection over Union* (IoU)

IoU digunakan untuk mengukur tingkat kesesuaian antara *Bounding Box* hasil prediksi dengan ground truth (posisi sebenarnya dari objek) [35]. IoU merupakan rasio antara luas area irisan (overlap) dan luas area gabungan (union) dari kedua kotak tersebut.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{15}$$

Jika nilai IoU lebih besar atau sama dengan ambang batas tertentu (misalnya 0,5), maka deteksi dianggap benar (*True Positive*). IoU menjadi dasar utama dalam perhitungan *mean Average Precision* (mAP).

5. *Mean Average Precision* (mAP)

mAP merupakan metrik utama yang digunakan untuk menilai performa keseluruhan model deteksi objek. Nilai ini didapatkan dari rata-rata Average *Precision* (AP) semua kelas objek yang terdeteksi [35].

Average *Precision* (AP) sendiri merupakan luas area di bawah kurva *Precision–Recall* (PR Curve) untuk satu kelas tertentu. Kurva ini menggambarkan hubungan antara nilai *Precision* dan *Recall* pada berbagai nilai ambang batas (confidence threshold) [32]. Berikut rumus AP:

$$AP = \int_0^1 P(R)dR \tag{16}$$

Selanjutnya, *mAP* dihitung dengan rata-rata nilai AP dari seluruh kelas yang ada:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{17}$$

P menunjukkan *Precision*, R menunjukkan *Recall*, dR perubahan kecil pada *Recall*, yaitu seberapa banyak *Recall* meningkat antara dua titik perhitungan. Kemudian N menunjukkan jumlah kelas objek dan AP_i menunjukkan Average *Precision* untuk kelas ke-i.

Dalam model YOLOv11, terdapat dua varian nilai mAP yang umum digunakan [35], yaitu:

a) mAP@0.5 artinya menghitung rata-rata presisi ketika threshold IoU ditetapkan sebesar 0,5, artinya deteksi dianggap benar jika *Bounding Box* hasil prediksi memiliki tumpang tindih minimal 50% dengan ground truth.
b) mAP@0.5:0.95 → menghitung rata-rata mAP pada berbagai nilai ambang IoU mulai dari 0,5 hingga 0,95 dengan interval 0,05.

Metrik ini lebih ketat karena menilai performa model di berbagai tingkat toleransi kesesuaian posisi. Semakin tinggi nilai mAP@0.5:0.95, semakin baik kemampuan model dalam mengenali dan melokalisasi objek secara akurat.

2.11.6 Pengujian Realtime Model

Pada tahap pengujian *realtime*, model yang telah dilatih tidak hanya dievaluasi berdasarkan tingkat presisi, *recall*, dan *mean average precision (mAP)*, tetapi juga dari segi kecepatan pemrosesan saat dijalankan secara langsung. Metrik yang umum digunakan untuk mengukur performa ini adalah *Frames per Second* (FPS) [37] [33]. *Frames per Second* (FPS) menunjukkan jumlah *frame* yang dapat diproses model dalam satu detik. Semakin tinggi nilai FPS, semakin cepat sistem mendeteksi dan menampilkan hasil deteksi secara langsung. FPS dihitung menggunakan rumus:

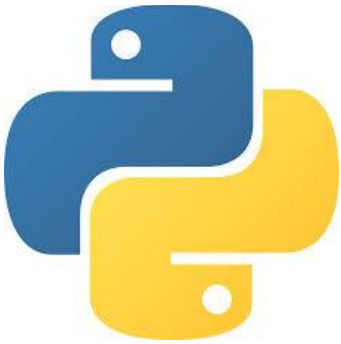
$$FPS = \frac{1}{t_{frame}} \quad (18)$$

t_{frame} merupakan waktu pemrosesan satu *frame* dalam hitungan detik. Kecepatan umum realtime suatu system yaitu lebih dari 15 fps, sehingga jika sudah lebih dari 15 fps sudah dikatakan realtime [38].

2.12 Python

Python merupakan bahasa pemrograman *open-source* yang populer dan serbaguna, digunakan baik untuk aplikasi mandiri maupun untuk skrip dalam berbagai bidang pengembangan perangkat lunak. Bahasa ini bersifat gratis, portabel, kuat, serta mudah dipelajari dan digunakan, menjadikannya pilihan strategis bagi pengembang di berbagai industri [39]. Python meruapakan bahasa yang sederhana dan mudah bagi pengguna. Untuk menjalankan program Python, pengguna cukup menulis dan mengeksekusinya secara langsung tanpa proses kompilasi dan linking seperti pada C atau C++. Hal ini memberikan pengalaman pemrograman yang interaktif serta memungkinkan pengujian dan perbaikan cepat. Karena sintaksnya yang sederhana dan menyerupai bahasa alami, Python sering disebut sebagai "*executable pseudocode*" yang dapat dibaca bahkan oleh non-programmer.

Selain kemudahan sintaks, Python memiliki pustaka standar yang luas serta dukungan ekosistem eksternal yang besar. Pustaka ini mencakup berbagai bidang, mulai dari manipulasi string, pemrosesan data, jaringan, grafika, hingga kecerdasan buatan. Komunitas open source Python juga terus menambah berbagai third-party utilities yang memperluas kemampuannya [39]. Python dapat digunakan dalam berbagai kebutuhan seperti analisis data, pembuatan model, pembuatan game, website, dan berbagai kebutuhan lainnya.



Gambar 25. Logo Python

Bahasa pemrograman Python merupakan salah satu bahasa yang paling banyak digunakan dalam pengembangan model *Machine Learning* dan *Deep Learning*. Hal ini disebabkan oleh sifat Python yang sederhana, fleksibel, dan memiliki ekosistem pustaka (library) yang sangat kaya, sehingga memudahkan proses pemodelan, pelatihan, dan evaluasi sistem berbasis kecerdasan buatan (AI).

2.13 Google Collab

Google Colab (Google Collaboratory) merupakan platform berbasis web yang memungkinkan pengguna menulis dan menjalankan kode Python langsung melalui peramban tanpa perlu melakukan instalasi perangkat lunak tambahan [40]. Pengguna tidak perlu menyiapkan lingkungan pemrograman secara lokal, yang umumnya memerlukan waktu dan konfigurasi teknis yang cukup rumit. Cukup dengan akun Google dan koneksi internet, pengguna dapat langsung mengakses lingkungan Python lengkap dan mulai melakukan pemrograman. Google colab dikembangkan oleh google yang bertujuan untuk mendukung penggunaan dalam analisis data, *Machine Learning*, atau hal lain yang dapat dilakukan dengan bahasa python.

Gambar 26. Logo Google Colab

Selain itu, Google Colab juga menyediakan fasilitas komputasi awan (cloud computing) yang memungkinkan pengguna menjalankan analisis data skala besar tanpa membutuhkan perangkat keras berperforma tinggi [40]. Salah satu keunggulan utamanya adalah tersedianya akses gratis ke sumber daya komputasi seperti GPU (*Graphics Processing Unit*) dan TPU (*Tensor Processing Unit*), yang sangat bermanfaat dalam mempercepat proses pelatihan model berbasis *Deep Learning*.

2.14 Github

GitHub merupakan sebuah platform penyimpanan dan kolaborasi proyek perangkat lunak yang berbasis pada sistem kontrol versi Git. Platform ini berfungsi untuk menyimpan, mengelola, serta mengoordinasikan kode sumber dari proyek perangkat lunak secara daring (online) [41]. Melalui GitHub, pengembang dapat melakukan kolaborasi dalam menulis, memperbarui, dan memantau perubahan kode dalam repositori secara efisien.

Gambar 27. Logo Github

GitHub didirikan pada tahun 2008 oleh Tom Preston-Werner, Chris Wanstrath, dan PJ Hyett. Gagasan pendiriannya berawal ketika Tom Preston-Werner mengusulkan konsep pembuatan situs web untuk berbagi repositori Git pada 18 Oktober 2007, saat menghadiri pertemuan komunitas programmer Ruby bersama Chris Wanstrath yang kemudian tertarik dan bergabung dalam pengembangannya. Pada tahap awal, Tom berfokus pada perancangan antarmuka pengguna serta komponen akses repositori menggunakan bahasa Ruby, sementara Chris mengembangkan aplikasi dengan kerangka kerja Ruby on Rails. Setelah proses pengembangan selama tiga

bulan, GitHub diperkenalkan dalam versi closed beta pada Januari 2008 dan hanya diakses melalui undangan, hingga kemudian PJ Hyett bergabung pada Februari 2008 untuk memperkuat tim. GitHub akhirnya resmi diluncurkan untuk publik pada 10 April 2008 dan berkembang pesat menjadi salah satu platform kolaborasi kode terbesar di dunia, digunakan oleh pengembang individu, perusahaan, dan komunitas open source untuk menyimpan, mengelola, serta mengembangkan perangkat lunak secara kolaboratif [41]. Dengan berbasis pada sistem kontrol versi Git, GitHub menjadi bagian penting dalam manajemen proyek perangkat lunak modern karena mampu menyimpan perubahan kode dan memfasilitasi kolaborasi lintas tim secara efisien.

2.15 Penelitian Terdahulu

Penelitian ini disusun dengan mengacu pada berbagai penelitian sebelumnya yang membahas deteksi objek, pengenalan bahasa isyarat, dan penggunaan model deep learning pada sistem *real-time*. Berbagai studi tersebut membantu penulis memahami konsep yang digunakan, melihat kekurangan dari penelitian sebelumnya, serta menemukan ide baru yang kemudian diterapkan dalam penelitian ini.

Tabel 2. Penelitian Terdahulu

No	Peneliti	Dataset	Metode	Hasil
1	Ahmed Mateen Buttar dkk (2023) [42]	video bahasa isyarat Amerika (ASL) yang direkam secara kustom dengan berbagai kondisi latar dan pencahayaan	YOLOv6 untuk deteksi tanda statis dan LSTM dengan MediaPipe untuk tanda dinamis	model YOLOv6 mencapai akurasi 96% untuk tanda statis dan LSTM mencapai 92% untuk tanda dinamis.

2	Soukaina Chraa Mesbahi, Mohamed Adnane Mahraz, Jamal Riffi, dan Hamid Tairi (2023) [32]	50.000 gambar tangan dari berbagai individu, kondisi pencahayaan, dan latar belakang, dengan tujuh kelas gerakan	YOLOv3, YOLOv4, YOLOv4-tiny, dan YOLOv5	YOLOv5 memberikan kinerja terbaik dengan akurasi mencapai 99,5% dan kecepatan pemrosesan hingga 155 FPS, melampaui versi YOLO sebelumnya
3	Osama A. Mohammed, Ammar Hawbani, dan Said H. Al-Hamadi (2023) [43]	Dataset publik yaitu MU HandImages ASL and OkkhorNama	Penerapan *attention module* pada YOLOv5	akurasi hingga 98,7% dan mempercepat waktu inferensi dibandingkan versi standar YOLOv5
4	Vipul Reddy P., Vishnu Vardhan Reddy B., dan Dr. Sukriti (2023)	Dataset yang digunakan berisi 288 citra warna dari 24 individu dengan kondisi pencahayaan bervariasi, mencakup 12 kelas gerakan	Algoritma YOLOv5	Model YOLOv5-medium yang dilatih selama 200 epoch menunjukkan performa terbaik dengan nilai *F1-Score* 90,5% dan *mean Average Precision (mAP)* sebesar 98,1%

5	Hizkia Halim dan Lina (2023)[44]	Perekaman video gerakan BISINDO dari teman-teman tuli dengan kamera beresolusi 1080p dan 720p, kemudian diekstraksi menjadi 1.500 citra	YOLOv5	Hasil validasi terbaik dicapai pada *epoch* 200 dengan *Precision*, *Recall*, dan *F1-Score* sebesar 1, sementara hasil pengujian menunjukkan akurasi tertinggi sebesar 67% untuk video satu kata dan 51,95% untuk video dua kata.
6	Ali Alzubaidi dan Khaled Elleithy (2024) [35]	Ribuan citra huruf ASL yang diambil dalam berbagai kondisi pencahayaan dan latar belakang	YOLOv8	Model YOLOv8 mencapai akurasi rata-rata 99,2% dan mampu mendeteksi isyarat secara cepat dengan *frame* rate tinggi, sehingga sesuai untuk aplikasi *Real-time*.
7	Wanjun Jia dan Changyong Li (2024) [45]	Dataset Bengali Sign Language Alphabet (BdSL)	Model SLR-YOLO	Model SLR-YOLO mencapai akurasi (mAP) sebesar 90.6% pada dataset American Sign Language Letters (ASLL) dan 98.5%

8	Melek Alaftekin, Ishak Pacal, dan Kenan Cicek (2024) [33]	Dataset berlabel yang terdiri dari 1500 gambar angka (0 hingga 9) dalam Bahasa Isyarat Turki	YOLOv4-CSP (Transformer)	Proposed YOLOv4-CSP mengungguli algoritma YOLO sebelumnya, mencapai *Precision* 98.95%, *Recall* 98.15%, F1 score 98.55%, dan mAP 99.49% dalam kecepatan 9.5 milidetik
9	Andhika Bayu Pangestu, Muhamad Rafi Muttaqin, dan Muhamad Agus Sunandar (2024) [46]	35 simbol BISINDO umum, yang dikumpulkan menggunakan webcam laptop dalam berbagai kondisi pencahayaan dan latar belakang	YOLOv8	YOLOv8 memiliki kinerja yang sangat baik dalam mendeteksi objek, dengan nilai *Precision* 0.958 (95.8%) , *Recall* 0.974 (97.4%) , dan mAP50 mencapai 0.995 (99.5%)
10	Rohit Raja, R Sreemathya, Mousami Turuk, Jayashree Jagdale, dan Mohammad	Dataset buatan sendiri bernama Image-ISL-PICT yang terdiri dari 3.580 gambar dari 115 kelas isyarat ISL, di	Membandingkan berbagai versi algoritma YOLO	Hasil komparasi menunjukkan bahwa YOLO NAS mengungguli versi lainnya, mencapai mAP@0.50 tertinggi sebesar

	Anis (2025) [34]	mana setiap kelas memiliki 20 hingga 40 gambar yang dilakukan oleh 10 penanda isyarat yang berbeda		0.982 untuk varian "S" dan 0.983 untuk varian "M", sementara versi YOLOv11 juga menunjukkan kinerja terbaik dengan mAP@0.50 sebesar 0.978

Penelitian-penelitian terdahulu telah menunjukkan bahwa berbagai versi YOLO mampu memberikan performa yang baik dalam mendeteksi bahasa isyarat secara *real-time*, mulai dari YOLOv4, YOLOv5, hingga YOLOv8. Tetapi dengan adanya model baru seperti YOLOv11 dapat menunjukkan peningkatan performa melalui arsitektur yang lebih efisien dan cepat. Karena itu, penelitian ini mengusulkan penggunaan YOLOv11 yang dikombinasikan dengan MediaPipe untuk meningkatkan akurasi sekaligus mempertahankan kecepatan deteksi. Selain itu, pengujian pada dataset SIBI yang berbeda dari penelitian-penelitian sebelumnya diharapkan dapat memberikan kontribusi baru dalam pengembangan sistem penerjemahan bahasa isyarat yang lebih adaptif dan optimal.

III. METODELOGI PENELITIAN

3.1 Waktu dan Tempat

Dalam menyelesaikan penelitian ini, telah dibuat jadwal pengerjaan serta tempat berlangsungnya kegiatan penelitian. Adapun rinciannya sebagai berikut:

Tempat :Laboratorium Teknik Komputer, Teknik Elektro, Fakultas Teknik, Universitas Lampung

Waktu : November 2025 – Maret 2026

Tabel 3. Waktu Penelitian

No	Aktivitas	Oktober 2025	November 2025	Desember 2025	Januari 2026
1	Identifikasi Masalah				
2	Kajian Pustaka dan Lapangan				
3	Pengumpulan Data				
4	*Preprocessing* Data				
5	Pelatihan Model				
6	Evaluasi Model				
7	Pengujian				

3.2 Tim Penelitian

Pengembangan proyek penelitian ini dilakukan bersama oleh tim yang terdiri dari satu orang pengembang frontend, satu orang pengembang backend, dan satu orang pembuat model AI. Rincian anggota tim beserta peran dan tugas masing-masing dapat dilihat pada tabel di bawah ini.

Tabel 4. Tim Penelitian

No	Nama	Role	Deskripsi Pekerjaan
1	Zaki Ahmad Basyary	Pembuat Model AI	• Mengembangkan dan melatih model AI menggunakan YOLOv11 dan MediaPipe untuk mendeteksi dan mengenali bahasa isyarat. • Melakukan data *preprocessing*, *training*, dan evaluation terhadap dataset gestur tangan. • Mengoptimalkan performa model agar akurat dan efisien untuk sistem *Real-time*.
2	Yusri Afta Putra	Frontend	• Mengembangkan frontend sistem menggunakan HTML, CSS, JavaScript, dan *frame*work frontend agar tampilan interaktif dan responsif. • Membuat antarmuka untuk menampilkan hasil prediksi dari model AI secara *Real-time*. • Mengoptimalkan tampilan agar dapat diakses melalui berbagai perangkat tapi menekankan dari sisi tampilan diperangkat mobile. • Berkolaborasi dengan backend developer dan AI developer dalam integrasi API.
3	Muhammad Faqih Himawan	Backend	• Membangun dan mengelola struktur backend menggunakan FastApi untuk mendukung fungsi utama sistem.

			• Mengembangkan API untuk komunikasi antara sistem dan model AI. • Mengelola database sistem • Memastikan keamanan dan efisiensi performa sistem di sisi server.

3.3 Alat Penelitian

Penelitian ini menggunakan perangkat dengan spesifikasi berikut:

3.3.1 Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan selama penelitian yaitu laptop dengan spesifikasi sebagai berikut:

Tabel 5. Tabel Perangkat Keras

No	Kebutuhan	Spesifikasi	Kegunaan
1	Laptop	AMD Ryzen 5 4600H GTX 1650 Ram 16GB	Perangkat keras yang digunakan dalam keseluruhan penelitian, seperti data analisis, data *preprocessing*, data modelling, dan data evaluation hingga penyusunan laporan.
2	Webcam	Logitech Resolusi 720p	Perangkat ini digunakan untuk pengujian *Real-time* model mandiri.
3	Komputer (AIO) ACER Veritonz4	Intel Core i7, dengan Ram 8, OS Windows 11 pro	Perangkat keras yang digunakan dalam pengujian *Real-time* model dengan responden

3.3.2 Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan pada penelitian sebagai berikut:

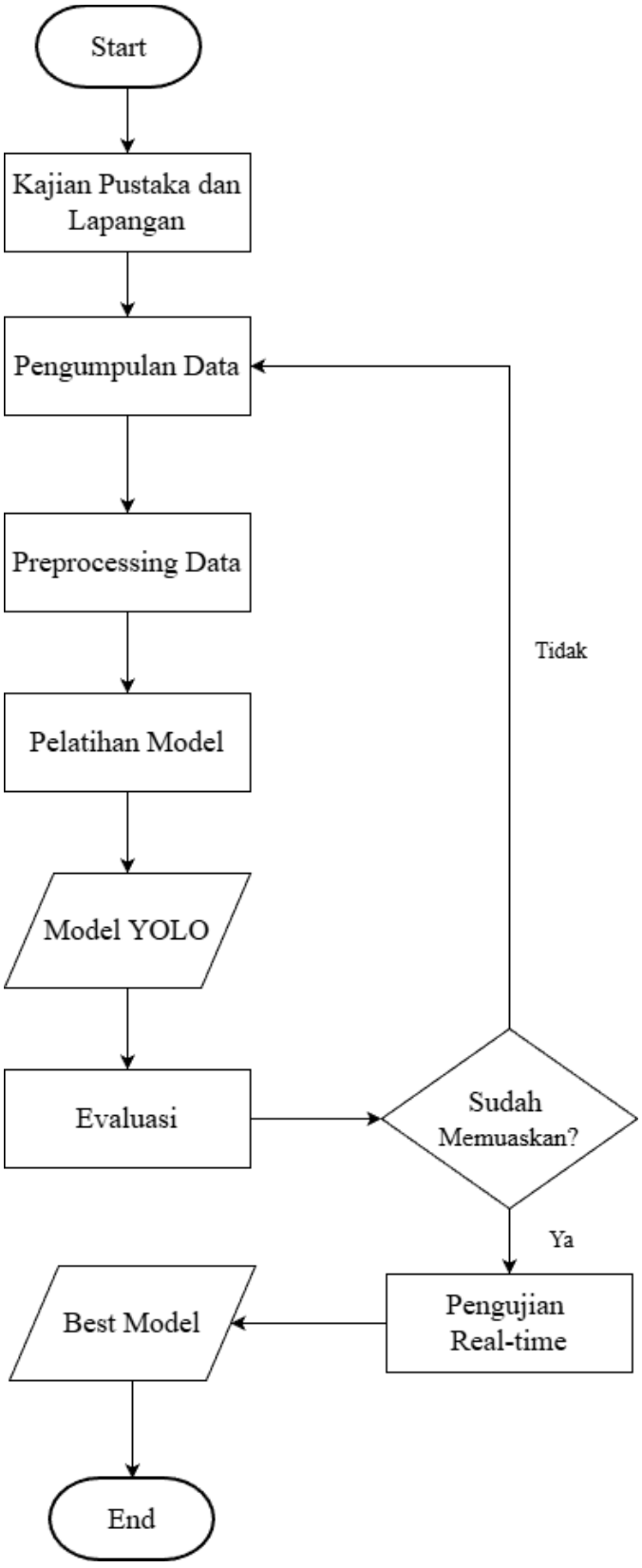
Tabel 6. Tabel Perangkat Keras

No	Kebutuhan	Versi	Kegunaan
1	Python	3.11.2	Bahasa pemrograman utama untuk pengolahan data citra, pelatihan model deteksi bahasa isyarat menggunakan YOLOv11 dan MediaPipe, serta pengujian realtime
	a.Ultralytics YOLOv11	-	*Frame*work *object detection* untuk mendeteksi posisi dan bentuk tangan secara *Real-time*.
	b. MediaPipe	-	Library dari Google untuk pelacakan dan ekstraksi *Landmark* tangan yang digunakan dalam proses pengenalan gestur
	c. OpenCV	-	Library pengolahan citra untuk membaca *frame* dari kamera, menampilkan hasil deteksi, dan melakukan *preprocessing* gambar.
	d. NumPy	-	Library komputasi numerik dan manipulasi array multidimensi yang digunakan untuk pengolahan data citra dan perhitungan vektor
	e. Pandas	-	Library analisis data untuk menyusun dataset gestur tangan dan mengelola hasil eksperimen model
	f. Matplotlib	-	Library visualisasi data untuk menampilkan grafik hasil pelatihan seperti akurasi dan *loss*
	g.Scikit-learn	-	Library *Machine Learning* yang digunakan untuk *data preprocessing*, evaluasi model (*confusion matrix, classification report*), dan validasi hasil.

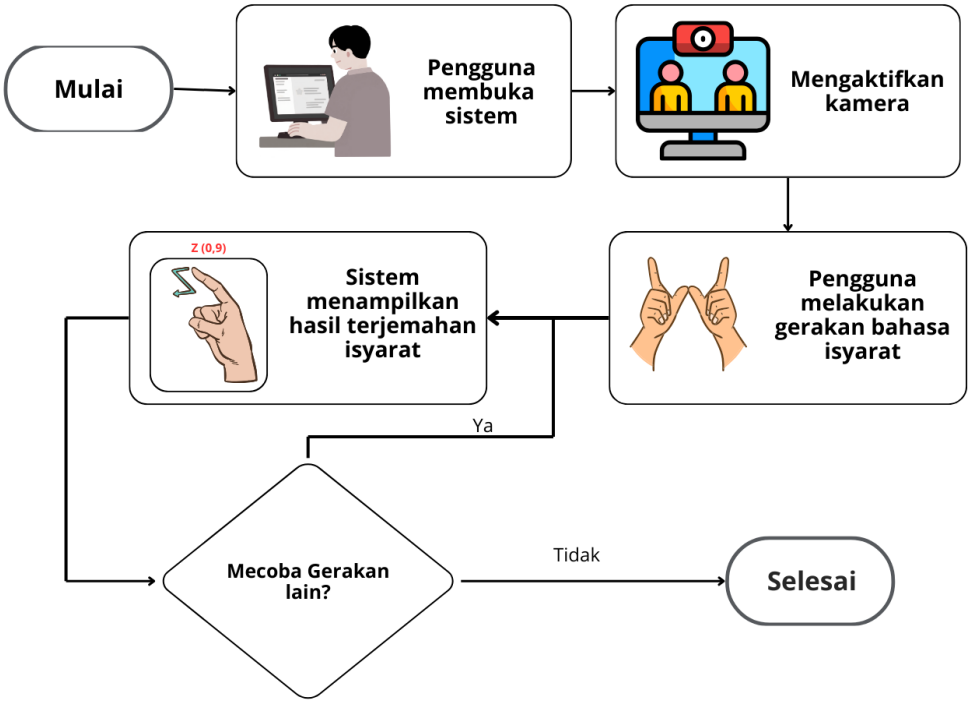
	h. PyTorch / TensorFlow	-	*Frame*work *Deep Learning* untuk eksperimen modifikasi model dan pelatihan ulang (fine-tuning) jaringan saraf.
2	Google Colaboratory	Website	Platform berbasis cloud untuk menjalankan kode Python, melatih model YOLOv11, dan melakukan eksperimen tanpa perlu instalasi lokal. Mendukung GPU untuk mempercepat proses *training*.
3	Git dan GitHub	Website	Digunakan untuk manajemen versi kode, kolaborasi antar anggota tim, dan penyimpanan repositori model AI.
4	Operating System	Windows 11 64-bit	Sebagai lingkungan perangkat lunak menjalankan model dan system lainnya.

3.4 Tahapan Penelitian

Penelitian ini dilakukan melalui beberapa tahapan yaitu meliputi studi literatur, pengumpulan data, *preprocessing* data, pelatihan data, evaluasi model, dan pengujian *realtime* model yang dimana tahapan ini memiliki referensi dari penelitian sebelumnya [34], [35], [36], Berikut ini adalah tahap-tahapan dalam penelitian ini untuk membangun model AI yang akan digunakan yaitu sebagai berikut:



Gambar 28. Tahapan Penelitian



Gambar 29. Alur Penggunaan Sistem

3.4.1 Kajian Pustaka dan Lapangan

Gambar 30. Studi Literatur

Pada tahap ini terdiri dari studi literatur serta observasi dan wawancara. Tahap studi literatur dilakukan melalui pengumpulan referensi dari berbagai sumber seperti buku, jurnal ilmiah, dan penelitian terdahulu yang relevan yang mendukung pengambangan system.Kegiatan yang dilakukan pada tahap studi literatur meliputi:

1. Mempelajari konsep dasar bahasa isyarat, terutama bentuk dan gerakan tangan yang digunakan dalam komunikasi visual.
2. Mempelajari teori terkait *Computer Vision* dan *Machine Learning*, khususnya model deteksi objek.
3. Mencari penelitian yang pernah dilakukan dibidang terkaiat dan menemukan metode yang digunakan serta permasalahan yang terjadi.
4. Menganalisis berbagai model dan metode yang dilakukan pada penelitian sebelumnya dan mempelajari model yang akan digunakan.

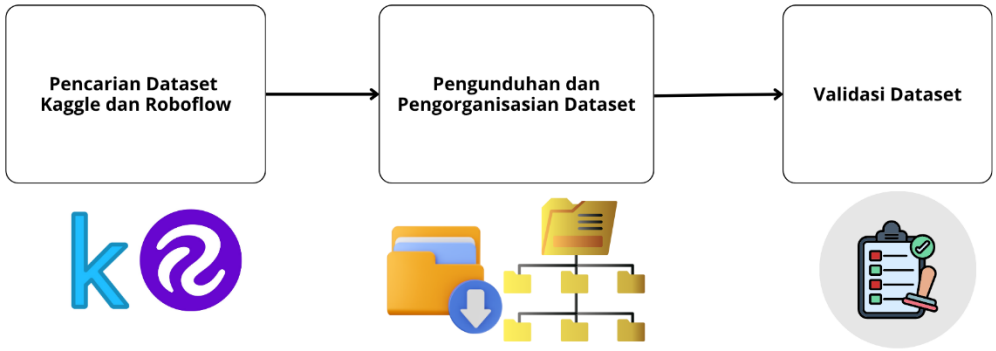
Pada tahapan ini selain studi literatur, dilakukan kegiatan observasi dan wawancara langsung. Observasi dan wawancara dilakukan di Sekolah Luar Biasa (SLB) Dharma Bhakti Dharma Pertiwi, yang merupakan salah satu lembaga pendidikan bagi siswa dengan hambatan pendengaran. Kegiatan ini bertujuan untuk memahami secara langsung permasalahan yang dihadapi dalam proses pembelajaran bahasa isyarat, termasuk metode pengajaran yang digunakan, tingkat kesulitan siswa dalam memahami gerakan tangan, serta kebutuhan akan media pembelajaran yang lebih interaktif dan adaptif.

Dari hasil studi literatur yang dilakukan, ditemukan bahwa beberapa penelitian sebelumnya masih menghadapi kendala dalam implementasi sistem pengenalan bahasa isyarat secara *Real-time* [47]. Permasalahan yang umum dijumpai meliputi penggunaan pada impelementasi *Real-time* dengan penggunaan komputasi yang tidak besar menjadi tantangan penelitian sebelumnya. Selain itu, dari hasil observasi dan wawancara, metode pembelajaran yang efektif yaitu menggunakan media yang interaktif sehingga baik siswa tunarungu maupun masyarakat umum lebih mudah mengenali bahasa isyarat.

Berdasarkan kebutuhan tersebut, penelitian diusulkan model YOLOv11 karena merupakan model yang lebih ringan dan efisien, memiliki proses inferensi yang lebih cepat, serta arsitektur yang telah dioptimalkan untuk *Real-time* [14]. YOLOv11 juga menyediakan beberapa varian ukuran model, yaitu n (nano), s (small), m (medium), dan l (large), yang memungkinkan penyesuaian antara kecepatan dan akurasi sesuai kebutuhan system sesuai kebutuhan sistem pembelajaran interaktif. Dengan karakteristik tersebut, YOLOv11 dinilai lebih

sesuai untuk diintegrasikan dalam media pembelajaran visual yang responsif dan mudah dipahami pengguna.

3.4.2` Pengumpulan Data



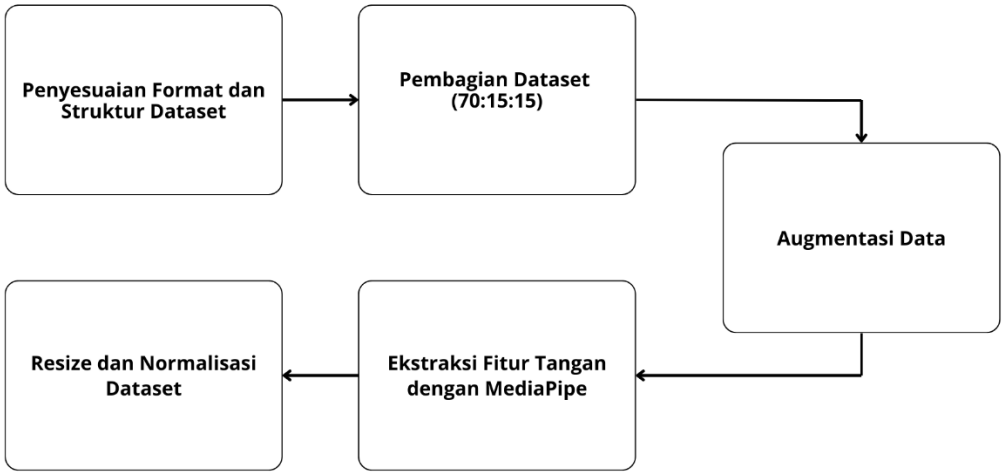
Gambar 31. Pengumpulan Dataset

Pada tahap ini, proses pengumpulan data dilakukan yang digunakan dalam pelatihan model pengenalan bahasa isyarat. Dataset terdiri dari 26 huruf alfabet (A–Z) dan 9 kata umum yang sering digunakan dalam komunikasi sehari-hari seperti saya, kamu, maaf, dan sebagainya. Setiap kelas memiliki minimal 50 citra berbeda, yang diambil dari beberapa individu dengan kondisi pengambilan gambar yang bervariasi.

Data diperoleh dari sumber publik yaitu Kaggle dan Roboflow, yang menyediakan kumpulan citra tangan berlabel berdasarkan huruf atau kata dalam bahasa isyarat. Untuk memastikan keberagaman data, citra yang dipilih memenuhi beberapa kriteria, antara lain:

1. Dataset terdiri memuat foto dari beberapa orang dengan bentuk tangan dan warna kulit yang berbeda sehingga model dapat belajar dari data yang beragam bentuk.
2. Variasi latar belakang mencakup latar polos dan non-polos agar model mampu beradaptasi dengan lingkungan nyata.
3. Perbedaan jarak dan posisi tangan terhadap kamera yaitu mencakup variasi jarak (dekat, sedang, jauh) serta posisi tangan yang sedikit berubah arah, seperti sedikit bergeser ke kanan maupun kiri, untuk meniru kondisi alami saat seseorang melakukan gerakan bahasa isyarat.

3.4.3 *Preprocessing* Data



Gambar 32. *Preprocessing* Data

Tahap ini dilakukan proses prapengolahan data, di mana data dari berbagai sumber seperti Kaggle dan Roboflow diseragamkan dan dibersihkan agar dapat digunakan secara optimal pada proses pelatihan model. Proses tersebut yaitu meliputi penyesuain format dan struktur dataset, *augmentasi* data, ekstraksi fitur dengan mediapipe, pembagian dataset, serta *resize* dan normalisasi data.

Langkah-langkah pemrosesan data yang dilakukan adalah sebagai berikut:

A. Penyesuaian Format dan Struktur Dataset

Pada tahap *preprocessing* dataset, dilakukan penyesuaian format dan struktur data untuk memastikan kesesuaian dengan kebutuhan proses *preprocessing* dan pelatihan model. Setiap file citra diberi format penamaan yang seragam, yaitu label_001.png, label_002.png, dan seterusnya. Penyeragaman penamaan ini bertujuan untuk mempermudah proses *preprocessing*, khususnya pada tahap pembuatan bounding *box*, sehingga nama file citra dapat sesuai dengan nama file label anotasi. Selain itu, dataset disusun dalam struktur folder terpisah berdasarkan kelas. Setiap kelas disimpan dalam satu folder khusus yang dinamai sesuai dengan label kelasnya, seperti A, B, C, Saya, Maaf, dan kelas lainnya. Setiap folder berisi seluruh citra yang merepresentasikan kelas tersebut. Struktur penyimpanan ini diterapkan untuk memudahkan pengelolaan data serta memperlancar tahapan *preprocessing* dan pelatihan model pada tahap selanjutnya.

B. Pembagian Dataset

Pada tahap ini, dataset dibagi menjadi tiga bagian, yaitu data latih (*training*), data validasi (*validation*), dan data uji (*testing*) dengan proporsi 70%, 15%, dan 15%.

1. Data latih (70%) digunakan untuk melatih model agar dapat mengenali pola dari data. Pada tahap ini, model belajar menyesuaikan bobot dan parameter internalnya berdasarkan contoh-contoh gambar yang diberikan.
2. Data validasi (15%) berfungsi untuk memantau kinerja model selama proses pelatihan. Data ini tidak digunakan untuk memperbarui bobot, tetapi digunakan untuk mengevaluasi apakah model mengalami overfitting atau tidak, sehingga membantu dalam menentukan parameter terbaik.
3. Data uji (15%) digunakan setelah pelatihan selesai, untuk mengukur sejauh mana model dapat melakukan generalisasi terhadap data baru yang belum pernah dilihat sebelumnya.

C. *Augmentasi* Data

Setelah dilakukan pembagian dataset, dilakukan augmentasi pada dataset *train*. Augmentasi data hanya diterapkan pada data latih (*training* set) untuk meningkatkan variasi data dan mencegah overfitting, sedangkan data validasi dan pengujian tidak mengalami augmentasi agar evaluasi performa model tetap objektif dan merepresentasikan kondisi data asli. Teknik *augmentasi* yang digunakan meliputi teknik transformasi geometris dan teknik transformasi warna dan intensitas. Pada kategori transformasi geometris, dilakukan proses rotasi sebesar ±15° ke arah kanan maupun kiri. Teknik ini bertujuan untuk meniru variasi sudut pengambilan gambar yang sering terjadi secara alami, misalnya ketika posisi kamera sedikit bergeser atau pengguna tidak selalu mempertahankan sudut tangan yang sama. Dengan adanya variasi ini, model dilatih untuk tetap mengenali gestur meskipun sudut pandangnya berubah. Selain rotasi, diterapkan pula flipping horizontal yang menghasilkan citra pantulan dari sisi kiri ke kanan. Teknik ini penting karena gestur tangan dapat dilakukan oleh tangan kanan maupun kiri, atau terekam dari arah yang berbeda, sehingga flipping membantu model menjadi lebih adaptif terhadap variasi orientasi dan posisi tangan pada kenyataan.

Sementara itu, pada kategori transformasi warna dan intensitas, dilakukan penyesuaian tingkat kecerahan (brightness adjustment). Proses ini mensimulasikan berbagai kondisi pencahayaan yang mungkin muncul pada saat pengambilan gambar, seperti lingkungan yang terlalu terang, redup, atau mengalami bayangan tertentu. Variasi pencahayaan seperti ini sangat umum terjadi dalam penggunaan *real-time*, terutama ketika sistem digunakan di ruang kelas, rumah, atau area terbuka dengan intensitas cahaya yang tidak selalu stabil. Dengan melakukan penyesuaian kecerahan, model dapat belajar untuk tetap mengenali gestur secara konsisten meskipun pencahayaan berubah.

Melalui *augmentasi* data, setiap kelas yang sawalnya memiliki minimal 50 citra diperbanyak hingga mencapai sekitar 200 citra per kelas. Sehingga jumlah keseluruhan dataset meningkat secara signifikan, dan model diharapkan dapat belajar lebih baik terhadap variasi kondisi nyata seperti perubahan arah tangan, sudut pandang kamera, maupun intensitas cahaya.

D. Ekstraksi Fitur dengan MediaPipe

Setelah proses *augmentasi*, setiap gambar baik pada dataset *train, validation,* maupun *test* diproses menggunakan MediaPipe *Hands*. Mediapipe akan mendeteksi keberadaan tangan dengan mengidentifikasi lokasi *keypoint* tangan *(landmark*) dan area bounding yang muncul pada citra. Output dari proses ini berupa koordinat tiga dimensi **(x, y, z)** untuk setiap titik *Landmark*. Nilai x dan y menunjukkan posisi relatif dari setiap titik terhadap lebar dan tinggi gambar (dalam skala 0–1), sedangkan nilai z menunjukkan kedalaman atau jarak relatif tangan terhadap kamera. Selanjutnya, dari sekumpulan titik tersebut, sistem menghitung titik ekstrem yaitu nilai minimum dan maksimum dari koordinat x dan y untuk menentukan area yang mengurung tangan sepenuhnya.

Nilai-nilai ini digunakan untuk membentuk bounding *box*, yang direpresentasikan dalam empat parameter utama:

1. Label : kelas dari gambar tersebut
2. x_center : titik tengah dari sumbu x,
3. y_center : titik tengah dari sumbu y,
4. width : lebar kotak pembatas,

5. *height* : tinggi kotak pembatas.

Bounding Box yang telah dihasilkan kemudian dikonversi ke format label YOLO, dengan struktur: <label> <*class*_id> <x_center> <y_center> <width> <*height*>. Seluruh nilai disimpan dalam bentuk rasio (0–1) terhadap ukuran gambar. File label disimpan dengan nama yang sama dengan citra aslinya. Dengan adanya *Bounding Box* yang presisi, model YOLO dapat mempelajari lokasi tangan secara akurat, sehingga meningkatkan kemampuan sistem dalam mengenali posisi dan pola gerakan bahasa isyarat secara efektif. Nilai *Bounding Box* tersebut akan disimpan dalam file label berformat txt, yang dimana nama file label tersebut akan sama dengan nama dari file foto yang dilabeli seperti A_001.png maka file label akan bernama A_001.txt.

E. *Resize* dan Normalisasi Dataset

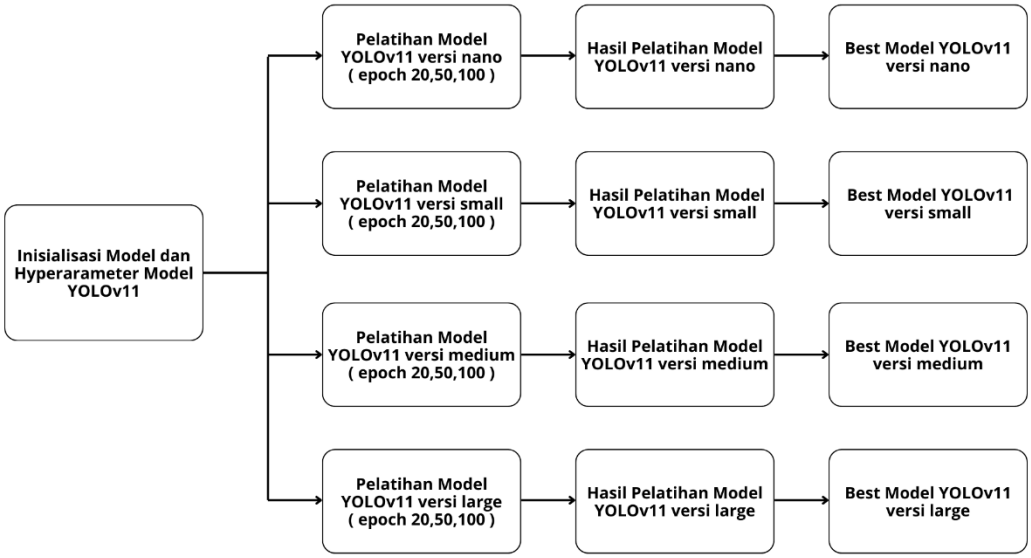
Tahap selanjutnya adalah melakukan penyesuaian data agar seluruh citra siap digunakan dalam proses pelatihan model. Proses *resize* dilakukan secara otomatis oleh YOLOv11 saat pelatihan dimulai, di mana seluruh citra diubah ukurannya menjadi 640×640 piksel. Ukuran ini merupakan standar input yang direkomendasikan oleh pengembang YOLOv11 karena mampu menjaga keseimbangan antara kualitas citra dan efisiensi komputasi, sehingga model dapat mengenali objek dengan optimal tanpa membebani sumber daya secara berlebihan.

Pada model YOLOv11, model ini melakukan proses *resize* dengan menyesuaikan ukuran gambar asli ke ukuran yang dibutuhkan menggunakan teknik letter*box*ing. Teknik ini menjaga perbandingan ukuran gambar (aspect ratio) tetap sama dengan cara menambahkan bagian kosong berwarna hitam di pinggir gambar jika diperlukan, sehingga bentuk objek tidak berubah atau terdistorsi saat diperbesar atau diperkecil. Selain itu, YOLOv11 menggunakan metode bili*near* interpolation untuk membuat hasil *resize* tetap halus dan tidak pecah. Dengan cara ini, seluruh gambar memiliki ukuran yang seragam tanpa mengubah bentuk objek di dalamnya, sehingga model dapat mengenali pola dengan lebih baik selama proses pelatihan.

Selain itu, YOLOv11 juga melakukan normalisasi nilai piksel secara otomatis dengan mengubah rentang nilai piksel dari 0–255 menjadi 0–1 menggunakan teknik

Min–Max normalization. Pada teknik ini, setiap nilai piksel dibagi dengan nilai maksimum 255 sehingga seluruh piksel berada pada skala yang sama. Proses ini penting karena nilai piksel yang terlalu besar dapat membuat perhitungan di dalam jaringan menjadi tidak stabil. Dengan normalisasi Min–Max, data yang masuk ke model memiliki skala yang konsisten, sehingga proses pembelajaran menjadi lebih stabil, model dapat belajar lebih cepat, dan risiko terjadinya kesalahan seperti vanishing gradient dapat berkurang. Normalisasi ini juga membantu model memahami pola pada citra secara lebih efisien karena nilai piksel sudah berada dalam rentang yang lebih sederhana dan mudah diproses.

3.4.5 Pelatihan Model



Gambar 33. Pelatihan Model

Pelatihan model dilakukan menggunakan YOLOv11 dengan beberapa versi, yaitu n (nano), s (small), m (medium), dan l (large). Pemilihan beberapa versi model bertujuan untuk membandingkan performa masing-masing model berdasarkan ukuran jaringan, jumlah parameter, dan kompleksitas komputasi.

A. Hyperparameter

Berikut ini pengaturan parameter yang dilakukan pada pelatihan model yaitu sebagai berikut:

1. Batch Size

Pada pelatihan model YOLOv11 digunakan parameter batch size yaitu 16 karena keseimbangan memori GPU dan stabilitas pelatihan yang dimana jika batch size yang terlalu kecil dapat membuat proses *training* tidak stabil, sedangkan batch size terlalu besar membutuhkan memori GPU besar dan memperlambat update bobot. Dan juga pemilihan batch size 16 karena dengan ukuran ini komputasi menjadi efisien yang dimana dapat memproses citra memproses citra lebih cepat daripada batch kecil, tapi tetap berhasil melakukan pembelajaran tetap stabil. Beberapa penelitian terdahulu menggunakan batch size 16 karena ukurannya yang stabil dan cukup cepat [35], [43].

2. Epoch

Pada pelatihan model YOLOv11 digunakan parameter epoch yaitu 20, 50, dan 100. Hal ini bertujuan untuk membandingkan performa model dalam beberapa epoch pembelajaran [36]. Epoch 20 dapat digunakan untuk memberikan baseline awal performa model, melihat apakah model cepat belajar dari dataset. Lalu epoch Memberikan peluang model untuk belajar lebih banyak pola dari dataset tanpa terlalu lama, sehingga kemungkinan overfitting masih rendah. Dan epoch 100 digunakan untuk menguji batas kemampuan model dalam menyerap fitur dari data. Ini membantu mempelajari kapan model mulai mengalami overfitting, karena jika *loss validation* mulai naik sementara *loss training* turun, berarti model terlalu menyesuaikan diri pada data *training*.

3. Optimizer

Pada pelatihan model YOLOv11 digunakan optimizer yaitu adam dikarenakan adam efisien pada dataset kecil sampai menengah dam mampu memanfaatkan gradien dari batch kecil secara efektif sehingga sesuai dengan dataset bahasa isyarat yang tidak terlalu banyak . Kemudian beberapa penelitian menggunakan optimizer adam karena performanya yang lebih baik dari optimizer lainnya [33].

4. Learning Rate

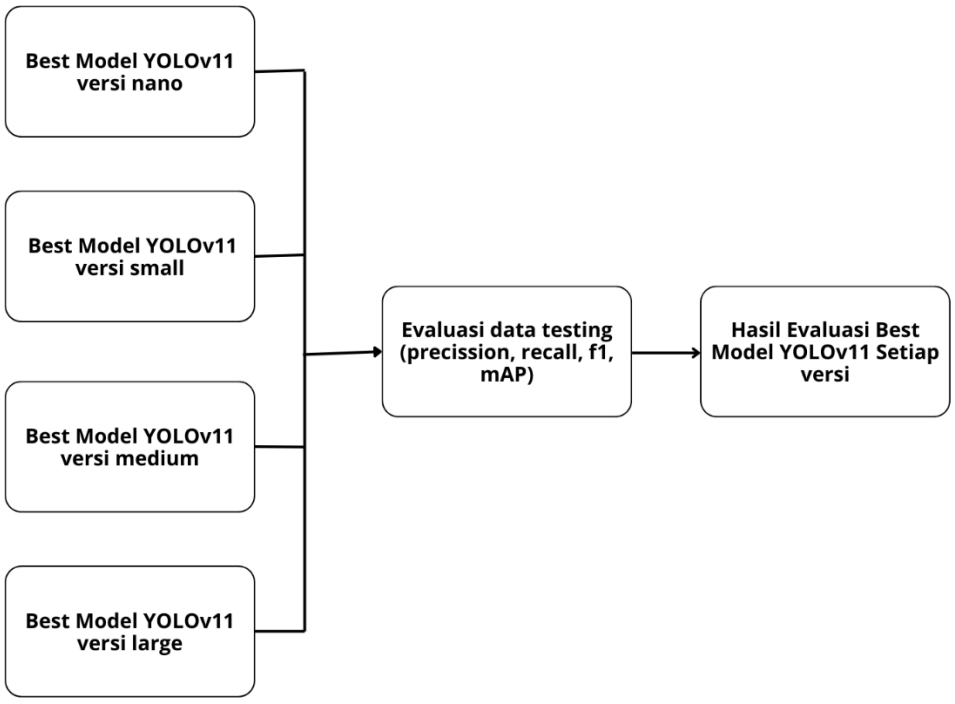
Pada pelatihan model YOLOv11 digunakan learning rate yaitu 0,001 dikarenakan learning rate kecil memungkinkan model menyesuaikan bobot secara halus, sehingga mempelajari fitur dari dataset bahasa isyarat dengan lebih akurat. Dan

juga learning rate 0,001 merupakan learning rate yang biasa digunakan ketika menggunakan optimizer adam.

B. Hasil Pelatihan Model

Setelah proses pelatihan model YOLOv11 dilakukan pada setiap versi model dan jumlah epoch yang telah ditentukan, diperoleh hasil evaluasi sementara berdasarkan data validasi (*validation* set). Data validasi digunakan untuk memantau performa model selama proses pelatihan serta menilai kemampuan generalisasi model terhadap data yang tidak dilibatkan dalam proses pembelajaran. Berdasarkan hasil evaluasi pada data validasi tersebut, dilakukan pemilihan model terbaik (best model) dari setiap versi YOLOv11. Pemilihan model terbaik ditentukan berdasarkan nilai metrik evaluasi yang dihasilkan pada data validasi, seperti mean Average *Precision* (mAP), *precision*, dan *recall*. Model dengan performa validasi terbaik dari masing-masing versi selanjutnya digunakan pada tahap evaluasi menggunakan data uji (*test* set) dan pengujian secara *real-time*.

3.4.6 Evaluasi Model



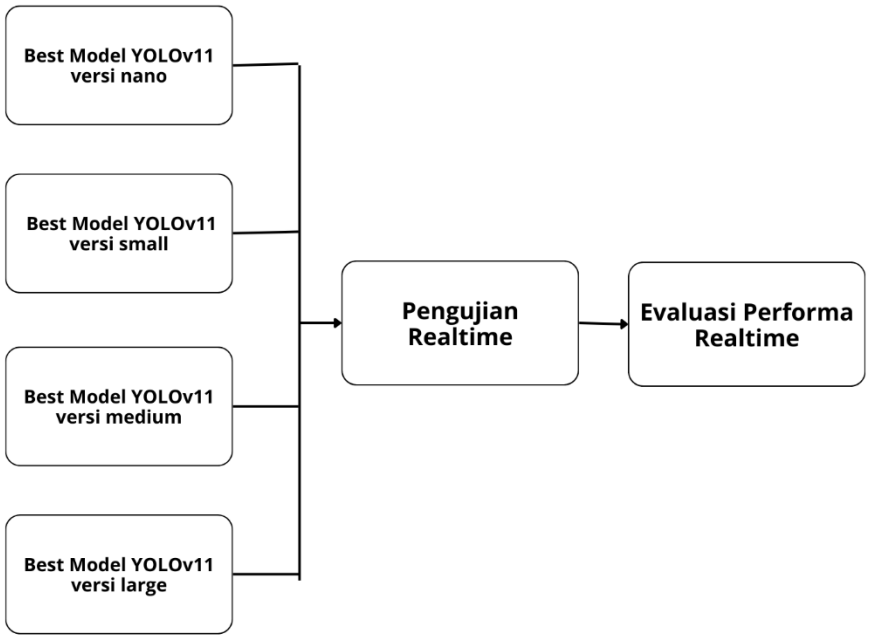
Gambar 34. Evaluasi Model

Model terbaik (best model) yang telah dipilih pada setiap versi YOLOv11 selanjutnya dievaluasi untuk menilai kinerjanya dalam mendeteksi bahasa isyarat

secara menyeluruh. Evaluasi dilakukan menggunakan dataset *test*, sehingga hasil yang diperoleh dapat merepresentasikan kemampuan model dalam menghadapi data yang belum pernah dilihat sebelumnya serta mendekati kondisi penggunaan nyata. Metrik evaluasi yang digunakan dalam penelitian ini meliputi *Precision*, *Recall*, F1-Score, dan mean Average *Precision* (mAP). Evaluasi dilakukan terhadap best model dari masing-masing versi YOLOv11, yaitu nano (n), small (s), medium (m), dan large (l). Pengujian ini bertujuan untuk membandingkan performa antar versi model berdasarkan hasil evaluasi data uji, sehingga dapat dianalisis pengaruh ukuran dan kompleksitas model terhadap kemampuan deteksi bahasa isyarat.

Hasil evaluasi tersebut selanjutnya digunakan sebagai dasar dalam menentukan model terbaik yang digunakan pada sistem penerjemah bahasa isyarat, yaitu model yang memiliki keseimbangan antara tingkat akurasi yang baik dan efisiensi komputasi yang memadai, sehingga mampu diimplementasikan secara optimal pada sistem berbasis *real-time*.

3.4.7 Pengujian



Gambar 35. Pengujian

Setelah proses evaluasi menggunakan data uji selesai dilakukan, setiap model terbaik (best model) yang telah ditentukan pada masing-masing versi YOLOv11 selanjutnya diuji secara *real-time* menggunakan kamera melalui OpenCV.

Pengujian ini bertujuan untuk menilai kinerja model dalam kondisi penggunaan langsung serta mengukur kemampuannya dalam mendeteksi bahasa isyarat secara responsif. Pada proses pengujian *real-time*, setiap *frame* video yang diterima dari kamera terlebih dahulu diubah ukurannya (*resize*) sesuai dengan ukuran input model, kemudian diproses untuk mendeteksi tangan, menghasilkan bounding *box*, label gestur, serta nilai confidence yang ditampilkan secara overlay pada tampilan video. Untuk memastikan hasil pengujian yang objektif dan representatif, pengujian dilakukan terhadap seluruh best model dari setiap versi YOLOv11 dengan melibatkan tiga pengguna berbeda. Masing-masing pengguna melakukan serangkaian gestur SIBI dalam kondisi lingkungan, perangkat, dan durasi pengujian yang sama. Pendekatan ini bertujuan untuk mengurangi bias pengujian serta mengevaluasi ketahanan model terhadap variasi bentuk tangan dan perbedaan gaya dalam melakukan gestur.

Pengujian ini akan menilai akurasi dan kecepatan inferensi model menggunakan metrik FPS (*Frame*s per Second), yang menunjukkan jumlah *frame* yang dapat diproses setiap detik. Akurasi dihitung berdasarkan jumlah gestur yang berhasil dikenali dengan benar oleh model dibandingkan total gestur yang diuji, sedangkan FPS dihitung dengan mengukur waktu pemrosesan setiap *frame* dan merata-ratakannya untuk seluruh video uji. Semakin tinggi akurasi, semakin tepat model dalam mengenali gestur, dan semakin tinggi FPS, semakin responsif model dalam mendeteksi bahasa isyarat secara *real-time*. Kedua metrik ini digunakan untuk memastikan bahwa sistem tidak hanya mampu mengenali gestur dengan benar, tetapi juga cukup cepat dan efisien saat digunakan secara langsung oleh pengguna.

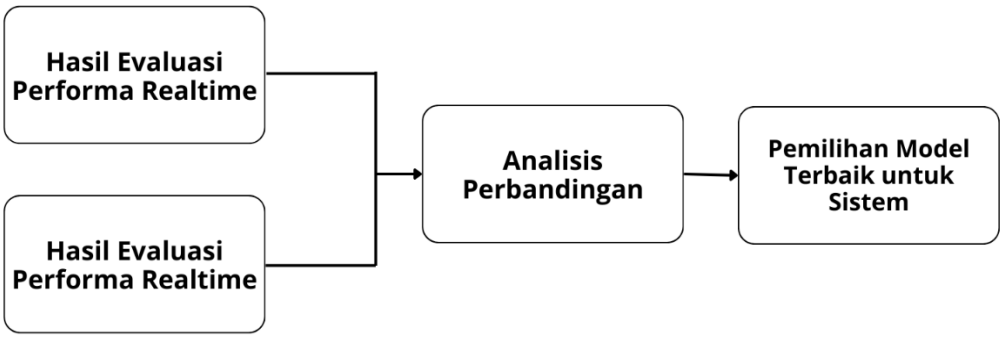
Berikut ini scenario pengujian *real-time* model sebagai berikut:

1. Lingkungan pengujian: Pengujian dilakukan di ruangan dengan menggunakan perangkat computer yang sama, dengan jarak pengguna ke kamera dibuat tetap, dan dengan durasi waktu yang sama.
2. Model dan kamera : model YOLOv11 dijalankan melalui Python dan kamera dibuka menggunakan OpenCV. Setiap *frame* otomatis di-*resize*, dinormalisasi, dan diproses untuk menghasilkan bounding *box*, label gestur, dan nilai confidence.

3. Pengujian oleh tiga pengguna : tiga pengguna melakukan gestur SIBI yang sama dengan jumlah repetisi dan durasi pengujian yang seragam, guna memastikan hasil yang adil dan representatif.
4. Parameter yang dicatat: parameter yang diukur meliputi akurasi prediksi, dan kecepatan pemrosesan (FPS) model dalam mengenali bahasa isyarat.
5. Evaluasi hasil : performa model dibandingkan berdasarkan hasil ketiga pengguna untuk menilai konsistensi akurasi dan kemampuan *real-time* model.

Hasil pengujian *real-time* pada best model setiap versi model selanjutnya digunakan sebagai salah satu dasar juga dalam menentukan model akhir yang digunakan pada sistem penerjemah bahasa isyarat, dengan mempertimbangkan keseimbangan antara tingkat akurasi deteksi, kestabilan performa, serta efisiensi komputasi agar sistem dapat berjalan secara optimal dalam skenario penggunaan nyata.

3.4.8 Pemilihan Model



Gambar 36. Pemilihan Model Untuk Sistem

Pemilihan model yang digunakan pada sistem penerjemah bahasa isyarat dilakukan berdasarkan hasil evaluasi menggunakan dataset *test* dan hasil pengujian *real-time*. Evaluasi menggunakan data *test* bertujuan untuk menilai kemampuan generalisasi model dalam mendeteksi bahasa isyarat pada data yang belum pernah dilihat sebelumnya, sedangkan pengujian *real-time* digunakan untuk menilai performa model dalam kondisi penggunaan langsung.

Pada tahap evaluasi data *test*, parameter yang digunakan sebagai dasar penilaian meliputi nilai *Precision*, *Recall*, F1-Score, dan mean Average *Precision* (mAP). Nilai-nilai tersebut digunakan untuk menilai ketepatan model dalam mengenali

gestur bahasa isyarat, keseimbangan antara deteksi benar dan kesalahan deteksi, serta performa deteksi secara keseluruhan. Hasil evaluasi data *test* ini digunakan sebagai salah satu dasar dalam menentukan model yang digunakan, dengan memastikan bahwa model memiliki tingkat akurasi dan kemampuan generalisasi yang memadai sebelum dan selama dilakukan pengujian *real-time*.

Pada penilaian realtime akan dilakukan menggunakan akurasi, yaitu seberapa banyak gestur yang berhasil dikenali dengan benar selama pengujian langsung. Pada tahap ini, penilaian difokuskan pada akurasi pengenalan gestur secara langsung, yaitu seberapa banyak gestur bahasa isyarat yang berhasil dikenali dengan benar selama proses pengujian menggunakan kamera. Penilaian ini menjadi penting karena mencerminkan kinerja model pada skenario penggunaan nyata.

Dengan demikian, model yang digunakan pada sistem penerjemah bahasa isyarat ditentukan berdasarkan kombinasi hasil evaluasi menggunakan dataset *test* dan hasil pengujian *real-time*. Model terpilih merupakan model yang memiliki kemampuan generalisasi dan tingkat akurasi yang baik pada data uji, serta mampu mempertahankan performa yang stabil pada pengujian langsung. Karena jumlah parameter (parameters) dan nilai FLOPs pada setiap versi model telah bersifat tetap, maka proses pemilihan model lebih difokuskan pada kemampuan model dalam mencapai standar kecepatan inferensi. Model dianggap memenuhi kriteria *real-time* apabila mampu mencapai kecepatan minimal 15 FPS sebagai standar kecepatan model [38], sehingga dapat memberikan respons yang cepat dan stabil ketika digunakan secara langsung oleh pengguna.

V. SIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, berikut adalah kesimpulan dari penelitian.

1. Model pengenalan bahasa isyarat SIBI berhasil secara optimal diimplementasikan dengan mengombinasikan YOLOv11 dan MediaPipe, dengan MediaPipe berperan dalam lokalisasi tangan dan YOLOv11 sebagai pendeteksi gestur. Performa yang optimal ditunjukkan dengan hasil akurasi lebih dari 90 % dan kecepatan realtime lebih dari 15 FPS.
2. Analisis hasil perbandingan performa antar versi model YOLOv11 pada tahap training dan testing, model YOLOv11 Nano mencapai performa terbaik ketika dijalankan pada spesifikasi GPU *Google Colaboratory*, dengan nilai mAP@0.5 sebesar 0,94655 pada training dan 0,936 pada testing, serta mAP@0.5:0.95 sebesar 0,77426 pada training dan 0,776 pada testing. Sementara itu, model dengan kompleksitas lebih tinggi menunjukkan kecenderungan penurunan nilai evaluasi.
3. Analisis hasil pengujian *real-time* menunjukkan model YOLOv11 Nano merupakan model yang paling sesuai ketika diuji pada spesifikasi perangkat berbasis CPU, karena mampu mencapai akurasi sebesar 97% dengan kecepatan inferensi rata-rata 18,63 FPS, sehingga mampu memenuhi kriteria real-time dibandingkan versi model lainnya yang memiliki kecepatan inferensi lebih rendah.

5.2 Saran

Saran yang dapat saya sampaikan dalam penelitian dikemudian hari adalah.

1. Dapat dilakukan peningkatan dengan menambah dataset dengan lebih banyak variasi subjek, sehingga model mampu mengenali perbedaan karakteristik gerakan antar pengguna. Dan juga penambahan noise pada data pelatihan juga

dapat diterapkan untuk meningkatkan ketahanan model terhadap kondisi lingkungan nyata, seperti variasi pencahayaan, latar belakang, dan gangguan visual lainnya.

2. Penelitian selanjutnya disarankan untuk melakukan analisis komputasi yang lebih mendalam melalui perbandingan kinerja sistem pada berbagai spesifikasi perangkat, baik berbasis CPU maupun GPU yang bertujuan untuk menganalisa pengaruh perbedaan kemampuan perangkat terhadap akurasi dan kecepatan inferensi sistem.
3. Pengembangan selanjutnya disarankan menggunakan dataset gestur dinamis dalam bentuk video atau rangkaian frame berurutan, serta menambahkan pemodelan informasi temporal agar sistem tidak hanya bergantung pada satu frame citra sehingga dapat merepresentasikan pergerakan tangan secara lebih baik dan meningkatkan akurasi pengenalan gestur dinamis secara keseluruhan.

DAFTAR PUSTAKA

[1] Kompasiana.com, “Fakta Penting Tentang Penyandang Disabilitas Rungu Wicara di Indonesia,” KOMPASIANA. Accessed: Nov. 06, 2025. [Online]. Available: https://www.kompasiana.com/duniatanpasuara/66e177a8ed6415119617f352/fakta-penting-tentang-penyandang-disabilitas-rungu-wicara-di-indonesia

[2] D. David *et al.*, “Sign language mobile apps: a systematic review of current app evaluation progress and solution framework,” *Evol. Syst.*, vol. 15, no. 2, pp. 669–686, Apr. 2024, doi: 10.1007/s12530-023-09494-0.

[3] S. Isnaniah, T. Agustina, Islahuddin, and F. Annisa, “The Use of Sign Language in Deaf Indonesian Classrooms in Surakarta,” *KEMBARA J. Sci. Lang. Lit. Teach.*, vol. 9, no. 2, pp. 468–481, Oct. 2023, doi: 10.22219/kembara.v9i2.25990.

[4] Z. Nikolawatin, P. Setyosari, and S. Ulfa, “Pengembangan Media Tutorial Bahasa Isyarat Untuk Siswa Tunarungu Slb Bc Kepanjen,” *JINOTEP J. Inov. Dan Teknol. Pembelajaran Kaji. Dan Ris. Dalam Teknol. Pembelajaran*, vol. 6, no. 1, pp. 15–22, Jul. 2019, doi: 10.17977/um031v6i12019p015.

[5] M. Sayuti and J. Pandawara, “Motion Graphic Media Pembelajaran Bahasa Isyarat Alfabet bagi Anak Tunarungu SDLB,” *Judikatif J. Desain Komun. Kreat.*, vol. 5, no. 1, pp. 14–21, Jun. 2023, doi: 10.35134/judikatif.v5i1.122.

[6] M. F. Wicaksono, M. D. Rahmatya, F. Ramadhanty, S. Tasya, and M. M. Iskandar, “Media Pembelajaran Huruf dan Angka dalam Bahasa Isyarat,” vol. 14, no. 1, 2025.

[7] T. Tao, Y. Zhao, T. Liu, and J. Zhu, “Sign Language Recognition: A Comprehensive Review of Traditional and Deep Learning Approaches, Datasets, and Challenges,” *IEEE Access*, vol. 12, pp. 75034–75060, 2024, doi: 10.1109/ACCESS.2024.3398806.

[8] M. Toshpulatov, W. Lee, J. Jun, and S. Lee, “Deep learning pathways for automatic sign language processing,” *Pattern Recognit.*, vol. 164, p. 111475, Aug. 2025, doi: 10.1016/j.patcog.2025.111475.

[9] R. S. L. Murali, “Sign Language Recognition System Using Convolutional Neural Network And ComputerVision,” *International Journal of Engineering Innovations in Advanced Technology*, vol. 4, no. 4, 2022.

[10] J. Bora, S. Dehingia, A. Boruah, A. A. Chetia, and D. Gogoi, “Real-time Assamese Sign Language Recognition using MediaPipe and Deep Learning,”

Procedia Comput. Sci., vol. 218, pp. 1384–1393, 2023, doi: 10.1016/j.procs.2023.01.117.

[11] L. A. Ranee, F. L. Marshillong, S. A. Lyndoh, and A. K. Maji, "Khasi Sign Language Recognition using Google's Mediapipe and Deep Learning Feedforward Neural Network Approach," *Procedia Comput. Sci.*, vol. 258, pp. 3619–3629, 2025, doi: 10.1016/j.procs.2025.04.617.

[12] S. Ingoley and J. Bakal, "Efficient Models Based on Deep Learning Technique for Indian Sign Language," *Procedia Comput. Sci.*, vol. 258, pp. 318–331, 2025, doi: 10.1016/j.procs.2025.04.269.

[13] W. Ismaiel, L. Kechiche, Y. Aribi, O. S. D. Omer, and W. Merghani, "Leveraging edge detection techniques to enhance Arabic sign language static-gesture recognition using deep learning," *J. Eng. Res.*, p. S2307187725005711, Sep. 2025, doi: 10.1016/j.jer.2025.09.011.

[14] A. A. Murat and M. S. Kiran, "A comprehensive review on YOLO versions for object detection," *Eng. Sci. Technol. Int. J.*, vol. 70, p. 102161, Oct. 2025, doi: 10.1016/j.jestch.2025.102161.

[15] A. M. Ambarak and A. Z. Falani, "Pengembangan Aplikasi Bahasa Isyarat Indonesia Berbasis Realtime Video Menggunakan Model Machine Learning," *JIKA J. Inform.*, vol. 7, no. 1, p. 89, Feb. 2023, doi: 10.31000/jika.v7i1.7277.

[16] E. Silpia and R. M. Sari, "Implementasi Komunikasi Bahasa Isyarat Anak Tunarungu," *JIIP - J. Ilm. Ilmu Pendidik.*, vol. 6, no. 1, pp. 529–535, Jan. 2023, doi: 10.54371/jiip.v6i1.1413.

[17] S. M. Ulfah and S. Ubaidah, "Penerapan Bahasa Isyarat dalam Pembelajaran bagi Anak Berkebutuhan Khusus Tuna Rungu," *J. Disabil. Stud. Res. JDSR*, vol. 2, no. 1, pp. 06–23, 2023.

[18] N. P. L. Wedayanti, A. P. Lintangsari, and G. A. P. Wirawan, "Perkembangan Bahasa Isyarat Daerah Denpasar," *Linguist. Indones.*, vol. 39, no. 2, pp. 217–223, 2021.

[19] S. S. Sindarto, D. E. Ratnawati, and I. Arwani, "Klasifikasi Citra Sistem Isyarat Bahasa Indonesia (SIBI) dengan Metode Convolutional Neural Network pada Perangkat Lunak berbasis Android," *J. Pengemb. Teknol. Inf. Dan Ilmu Komput.*, vol. 6, no. 5, pp. 2129–2138, 2022.

[20] Y. Zhang and X. Jiang, "Recent Advances on Deep Learning for Sign Language Recognition," *Comput. Model. Eng. Sci.*, vol. 139, no. 3, pp. 2399–2450, 2024, doi: 10.32604/cmes.2023.045731.

[21] Arnita, F. Marpaung, and F. Aulia, *Computer Vision Dan Pengolahan Citra Digital*. Surabaya: PUSTAKA AKSARA, 2022.

[22] Jamaaluddin and I. Sulistyowati, *Buku Ajar Kecerdasan Buatan (Artificial Intelligence)*. Sidoarjo: UMSIDA PRESS, 2021.

[23] Suyanto, *Artificial Intelligence : Searching, Reasoning, Planning, dan Learning*, 3rd ed. bandung: Informatika Bandung, 2021.

[24] S. Nurmaini, A. Darmawahyuni, and A. I. Sapitri, *Pengenalan Deep Learning dan Implementasinya*. Palembang: UPT. Penerbit dan Percetakan, 2021.

[25] B. Raharjo, *Deep Learning dengan Python*. Semarang: Yayasan Prima Agus Teknik, 2022.

[26] H. Kinsley and D. Kukieła, *Neural Networks from Scratch in Python*. Amerika Serikat: Kinsley Enterprises Inc, 2020.

[27] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd ed. Swiss: Springer, 2022.

[28] J. C. Lopes and R. P. Lopes, "Computer Vision in Augmented, Virtual, Mixed and Extended Reality environments—A bibliometric review," *Vis. Inform.*, vol. 8, no. 4, pp. 13–22, Dec. 2024, doi: 10.1016/j.visinf.2024.11.002.

[29] Kukil, "MediaPipe: Panduan Utama untuk Pemrosesan Video." Accessed: Dec. 06, 2025. [Online]. Available: https://learnopencv.com/introduction-to-mediapipe/

[30] B. Sundar and T. Bagyammal, "American Sign Language Recognition for Alphabets Using MediaPipe and LSTM," *Procedia Comput. Sci.*, vol. 215, pp. 642–651, 2022, doi: 10.1016/j.procs.2022.12.066.

[31] "Ultralytics YOLO11 - Ultralytics YOLO Docs." Accessed: Nov. 07, 2025. [Online]. Available: https://docs-ultralytics-com.translate.goog/models/yolo11/?_x_tr_sl=en&_x_tr_tl=id&_x_tr_hl=id&_x_tr_pto=tc

[32] S. C. Mesbahi, M. A. Mahraz, J. Riffi, and H. Tairi, "Hand Gesture Recognition Based on Various Deep Learning YOLO Models," *Int. J. Adv. Comput. Sci. Appl.*, vol. 14, no. 4, 2023, doi: 10.14569/IJACSA.2023.0140435.

[33] M. Alaftekin, I. Pacal, and K. Cicek, "Real-time sign language recognition based on YOLO algorithm," *Neural Comput. Appl.*, vol. 36, no. 14, pp. 7609–7624, May 2024, doi: 10.1007/s00521-024-09503-6.

[34] R. Raj, R. Sreemathy, M. Turuk, J. Jagdale, and M. Anish, "Performance Comparison of Different Versions of YOLO for Indian Sign Language Captioning in Real Time of Multiple Signers," *Procedia Comput. Sci.*, vol. 259, pp. 991–1000, 2025, doi: 10.1016/j.procs.2025.04.053.

[35] B. Alsharif, E. Alalwany, and M. Ilyas, "Transfer learning with YOLOV8 for real-time recognition system of American Sign Language Alphabet," *Frankl. Open*, vol. 8, p. 100165, Sep. 2024, doi: 10.1016/j.fraope.2024.100165.

[36] T. A. Jabar, R. Heriansyah, and E. Purnamasari, "Pengenalan Pola Huruf Bahasa Isyarat Menggunakan Framework You Only Look Once (YOLO)," *J. FASILKOM*, vol. 15, no. 2.

[37] R. Jalayer, M. Jalayer, C. Orsenigo, and M. Tomizuka, "A review on deep learning for vision-based hand detection, hand segmentation and hand gesture

recognition in human–robot interaction," *Robot. Comput.-Integr. Manuf.*, vol. 97, p. 103110, Feb. 2026, doi: 10.1016/j.rcim.2025.103110.

[38] M. Rivera-Acosta, J. M. Ruiz-Varela, S. Ortega-Cisneros, J. Rivera, R. Parra-Michel, and P. Mejia-Alvarez, "Spelling Correction Real-Time American Sign Language Alphabet Translation System Based on YOLO Network and LSTM," *Electronics*, vol. 10, no. 9, p. 1035, Apr. 2021, doi: 10.3390/electronics10091035.

[39] M. Lutz, *Learning Python*, 4th ed. Sebastopol: O'Reilly Media, 2009.

[40] N. Rofiq and S. L. M. Sitio, *Pengenalan Dasar Analisis Data Dengan Python di Google Colab*. Jawa Tengah: Eureka Media Aksara, 2024.

[41] H. Faqih, A. T. Pangestu, and C. Ramadani, *Modul Dasar Git dan Github*. Tegal: Universitas Bina Sarana Informatika, 2024.

[42] A. M. Buttar, U. Ahmad, A. H. Gumaei, A. Assiri, M. A. Akbar, and B. F. Alkhamees, "Deep Learning in Sign Language Recognition: A Hybrid Approach for the Recognition of Static and Dynamic Signs," *Mathematics*, vol. 11, no. 17, p. 3729, Aug. 2023, doi: 10.3390/math11173729.

[43] N. F. Attia, M. T. F. S. Ahmed, and M. A. M. Alshewimy, "Efficient deep learning models based on tension techniques for sign language recognition," *Intell. Syst. Appl.*, vol. 20, p. 200284, Nov. 2023, doi: 10.1016/j.iswa.2023.200284.

[44] H. Halim and L. Lina, "Aplikasi Pengidentifikasi Bahasa Isyarat Berdasarkan Gerak Tubuh Secara Real Time Menggunakan YOLO," *Simtek J. Sist. Inf. Dan Tek. Komput.*, vol. 8, no. 2, pp. 300–304, Oct. 2023, doi: 10.51876/simtek.v8i2.215.

[45] W. Jia and C. Li, "SLR-YOLO: An improved YOLOv8 network for real-time sign language recognition," *J. Intell. Fuzzy Syst.*, vol. 46, no. 1, pp. 1663–1680, Jan. 2024, doi: 10.3233/JIFS-235132.

[46] A. Bayu Pangestu, M. Rafi Muttaqin, and M. Agus Sunandar, "Sistem Deteksi Bahasa Isyarat Indonesia (BISINDO) Menggunakan Algoritma You Only Look Once (YOLO)V8," *JATI J. Mhs. Tek. Inform.*, vol. 8, no. 5, pp. 9891–9897, Sep. 2024, doi: 10.36040/jati.v8i5.10833.

[47] X. Jiang, Y. Zhang, J. Lei, and Y. Zhang, "A Survey on Chinese Sign Language Recognition: From Traditional Methods to Artificial Intelligence," *Comput. Model. Eng. Sci.*, vol. 140, no. 1, pp. 1–40, 2024, doi: 10.32604/cmes.2024.047649.