BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi dan komunikasi yang begitu cepat di masa kini telah berdampak pada berbagai aspek kehidupan manusia, termasuk cara orang berkomunikasi. Masalah besar yang masih terjadi adalah bagaimana memastikan setiap orang dapat berkomunikasi dengan lancar. Berkomunikasi merupakan hal yang penting bagi manusia dalam berinteraksi dengan orang lain, baik di rumah, sekolah, tempat kerja, maupun dalam lingkungan masyarakat (Wahyuni, 2025). Meskipun begitu, masih banyak orang yang merasa kesulitan dalam berbicara, seperti orang yang mengalami gangguan pendengaran atau kesulitan berkomunikasi.Akibatnya, bahasa isyarat adalah cara utama mereka berkomunikasi dan memberikan pesan (Soekarta et al., 2024).

Bahasa isyarat merupakan sistem komunikasi visual yang menggunakan gerakan tangan, ekspresi wajah, dan postur tubuh untuk menyampaikan informasi. Bahasa ini telah menjadi alat komunikasi vital bagi komunitas tunarungu dan tunawicara di seluruh dunia. Namun, tingkat pemahaman dan penguasaan bahasa isyarat di kalangan masyarakat umum masih sangat rendah. Hal ini menyebabkan terjadinya kesenjangan komunikasi antara individu dengan gangguan pendengaran sehingga seringkali menimbulkan diskriminasi, keterbatasan akses informasi, serta hambatan dalam memperoleh layanan publik yang setara (Celsia & Sandag, 2021).

Menurut data dari *(World Health Organization, 2025)* lebih dari 5% populasi global atau sekitar 430 juta orang mengalami gangguan pendengaran yang memerlukan rehabilitasi, termasuk di antaranya 34 juta anak-anak. Diperkirakan pada tahun 2050, jumlah ini akan meningkat menjadi lebih dari 700 juta orang, atau sekitar 1 dari 10 penduduk dunia. Fakta ini menunjukkan bahwa kebutuhan akan solusi teknologi yang dapat membantu proses komunikasi bagi individu dengan gangguan pendengaran semakin mendesak. Salah satu upaya yang dapat dilakukan adalah dengan mengembangkan sistem otomatis yang mampu mengenali dan mengklasifikasikan bahasa isyarat, khususnya angka, secara optimal dan efisien.

Seiring dengan kemajuan teknologi kecerdasan buatan *(Artificial Intelligence/AI)*, khususnya dalam bidang *computer vision* dan *deep learning*, peluang untuk menciptakan sistem pengenalan bahasa isyarat secara otomatis semakin terbuka lebar. Model-model *deep learning*, seperti *convolutional neural network (CNN)*, telah terbukti mampu melakukan klasifikasi gambar dengan performa yang baik. Dalam konteks pengenalan bahasa isyarat, CNN dapat digunakan untuk mengidentifikasi pola-pola visual yang terdapat pada gambar tangan yang membentuk angka atau huruf tertentu. Namun, pengembangan model *Deep Learning* yang efektif memerlukan jumlah data pelatihan yang besar dan beragam, serta proses pelatihan yang kompleks dan memakan waktu.

Untuk mengatasi keterbatasan data dan mempercepat proses pelatihan model, pendekatan *Transfer Learning* menjadi salah satu solusi yang banyak digunakan dalam penelitian-penelitian terkini. *Transfer Learning* memungkinkan pemanfaatan model yang telah dilatih pada *dataset* besar (seperti *ImageNet*) untuk diterapkan pada tugas

klasifikasi baru dengan jumlah data yang lebih sedikit. Dua arsitektur *CNN* yang populer dalam penerapan *Transfer Learning* adalah *VGG16* dan *ResNet50. VGG16* dikenal sebagai model yang sederhana namun efektif dalam klasifikasi gambar, sedangkan *ResNet50* merupakan model yang lebih dalam dengan 50 lapisan dan telah dilatih pada lebih dari satu juta gambar di database *ImageNet* (Berliani et al., 2023).

Berbagai penelitian telah dilakukan untuk menguji efektivitas *transfer learning* dalam klasifikasi bahasa isyarat dengan memanfaatkan teknologi *deep learning*. Penelitian oleh (Siddik, 2023) menunjukkan bahwa algoritma *Transfer Learning* pada arsitektur *VGG16* dan *ResNet50* mampu memberikan kinerja yang lebih baik dibandingkan metode tradisional dalam klasifikasi angka bahasa isyarat. Hal ini menunjukkan bahwa pemanfaatan model *Deep Learning* yang telah dilatih sebelumnya dapat meningkatkan efektivitas dan efisiensi proses pengenalan bahasa isyarat. Selain itu, penelitian oleh (Rachmawati et al., 2023) membuktikan bahwa model *Deep Learning* seperti *ResNet50, MobileNetV4, dan InceptionV3* dapat digunakan untuk klasifikasi Bahasa Isyarat Indonesia (BISINDO), dengan *ResNet50* menunjukkan performa yang sangat baik.

Di sisi lain, terdapat pula penelitian yang menggunakan pendekatan *non-deep learning*, seperti yang dilakukan oleh (Sugiantari et al., 2025). Penelitian ini menerapkan ekstraksi jarak *euclidean* antar landmark tangan menggunakan Mediapipe dan model *multi-layer perceptron (MLP)* untuk mengenali angka SIBI 0–9. Meskipun metode ini mampu bekerja secara real-time, performanya masih memiliki keterbatasan dalam menangani variasi pose tangan yang lebih kompleks, sehingga belum mampu mengungguli performa model *deep learning* yang lebih adaptif terhadap perubahan pola visual.

Meskipun berbagai penelitian sebelumnya telah membahas penggunaan *transfer learning* dalam pengenalan bahasa isyarat, masih terdapat kebutuhan untuk mengeksplorasi lebih lanjut bagaimana strategi pelatihan yang berbeda, khususnya yang berkaitan dengan variasi dan kuantitas data, dapat memengaruhi kinerja model dalam konteks klasifikasi angka bahasa isyarat. Variasi data pelatihan, seperti penggunaan data asli, data augmentasi, dan data gabungan (data asli dan data augmentasi), diyakini dapat memberikan dampak signifikan terhadap kemampuan generalisasi model. Selain itu, perbedaan skema pelatihan antara pelatihan langsung dari bobot ImageNet dan pelatihan lanjutan *(fine-tuning),* perlu dianalisis untuk bagaimana proses pembelajaran bertahap dapat mempengaruhi kinerja model dalam konteks klasifikasi angka bahasa isyarat.

Penelitian ini dilakukan dengan merujuk pada penelitian yang telah dilakukan oleh (Siddik, 2023) terkait klasifikasi angka bahasa isyarat menggunakan pendekatan *transfer learning*. Kebaruan *(novelty)* dalam penelitian ini adalah penggunaan variasi data melalui teknik augmentasi untuk melihat pengaruh data terhadap hasil kinerja model menggunakan pendekatan transfer langsung dan penerapan pelatihan lanjutan *(fine-tuning)* sebagai cara pembelajaran bertahap. Selain itu, penelitian ini mengintegrasikan pendekatan *Explainable AI (XAI)* dengan menggunakan metode *Grad-CAM (Gradient-weighted Class Activation Mapping)* untuk meningkatkan kemudahan dalam memahami hasil prediksi model.

Seluruh pendekatan tersebut kemudian di implementasikan ke dalam *dashboard* interaktif berbasis *Streamlit* tidak hanya menyajikan hasil prediksi secara *real-time*, tetapi juga memberikan informasi yang transparan melalui visualisasi *Grad-CAM*. Integrasi antara teknologi *transfer learning*, visualisasi *XAI*, dan *Dashboard* interaktif diharapkan dapat memberikan kontribusi nyata dalam mewujudkan komunikasi yang inklusif dan memberdayakan individu dengan gangguan pendengaran.

Berdasarkan uraian yang telah dijelaskan, maka peneliti memutuskan untuk melakukan penelitian dengan membandingkan kinerja metode *transfer learning* dalam mengklasifikasikan angka bahasa isyarat. Penelitian ini mempertimbangkan variasi *dataset* berupa data asli, data augmentasi, dan data gabungan (data asli dan data augmentasi) untuk menganalisis pengaruh variasi dan kuantitas data terhadap kinerja serta kemampuan generalisasi model. Selain itu, penelitian ini juga mengevaluasi pengaruh skema pelatihan independen dan pelatihan lanjutan *(fine-tuning)*. Hasil prediksi dari model yang dikembangkan kemudian diimplementasikan ke *Dashboard* sistem informasi berbasis *Streamlit*, yang dikembangkan untuk menyajikan hasil prediksi secara *realtime* dengan visualisasi menggunakan *Grad-CAM* secara interaktif dan informatif.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan sebelumnya, rumusan masalah yang akan dibahas diantaranya:

1. Bagaimana perbandingan kinerja antara model yang dirancang menggunakan pendekatan *Transfer Learning* langsung dengan model yang dilakukan pelatihan lanjutan (*fine-tuning*) dari model data asli terhadap data lainnya dalam mengklasifikasikan angka bahasa isyarat?
2. Bagaimana cara mengimplementasikan model prediksi dan visualisasi *Grad-CAM* ke dalam *dashboard* interaktif menggunakan *Streamlit?*

1.3 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Penelitian ini menggunakan dua arsitektur *Transfer Learning* utama, yaitu *VGG16* dan *ResNet50*, yang keduanya dimulai dari bobot *pre-trained ImageNet*
2. *Dataset* yang digunakan adalah *Hand Sign Language Digits* dari Kaggle.com, yang hanya terdiri dari 10 kelas angka (0-9).
3. Penelitian berfokus pada pembuatan model prediksi angka, tanpa memperluas ke gestur lain
4. Variasi *dataset* yang digunakan terbatas pada tiga jenis: Data Asli, Data Augmentasi, dan Data Gabungan (Asli + Augmentasi).
5. Penelitian ini tidak melakukan berbagai jenis augmentasi, hanya menggunakan rotasi untuk memperkaya variasi data

6. *Dashboard* sistem informasi dibangun menggunakan *Framework Streamlit*, sebagai antarmuka interaktif untuk menampilkan prediksi dan visualisasi *Grad-CAM*

1.4 Tujuan Penelitian

Tujuan pada penelitian ini adalah sebagai berikut:

1. Menganalisis dan membandingkan kinerja antara model yang dirancang menggunakan pendekatan *Transfer Learning* langsung dengan model yang dilakukan pelatihan lanjutan *(fine-tuning)* dari model data asli terhadap data lainnya dalam mengklasifikasikan angka bahasa isyarat.
2. Mengimplementasikan model prediksi beserta visualisasi *Grad-CAM* kedalam *Dashboard* interaktif berbasis *Streamlit*.

1.5 Manfaat Penelitian

Manfaat pada penelitian adalah sebagai berikut:

1. Memberikan kontribusi terhadap pengembangan metode klasifikasi citra, khususnya dalam penerapan *Transfer Learning* untuk pengenalan angka bahasa isyarat
2. Memberikan pemahaman yang lebih mendalam mengenai proses dan hasil prediksi angka bahasa isyarat menggunakan *Deep Learning*
3. Menghasilkan *Dashboard* sistem informasi menggunakan *Streamlit* yang dapat dimanfaatkan sebagai media visualisasi interaktif untuk menampilkan hasil prediksi model serta visualisasi *Grad-CAM*.

1.6 Landasan Teori

1.6.1 Klasifikasi Citra

Klasifikasi Citra merupakan proses dalam Pengolahan Citra Digital yang bertujuan untuk mengelompokkan sejumlah *pixel* atau *picture element* pada sebuah citra menjadi kelas-kelas. Setiap kelas ini mendeskripsikan suatu entitas yang memiliki karakteristik tertentu sehingga dapat dikenali (Herdiansah et al., 2022). Tujuan klasifikasi citra adalah mereplikasi kemampuan manusia dalam memahami informasi yang terkandung pada citra digital, sehingga komputer mampu mengenali dan mengelompokkan objek dalam citra sebagaimana manusia melakukannya. Tantangan utama dalam klasifikasi citra terletak pada proses feature engineering yang sering kali terbatas pada karakteristik *dataset* tertentu. Hal ini disebabkan oleh adanya variasi pada citra, seperti perbedaan sudut pengambilan, skala, kondisi pencahayaan, deformasi objek, serta faktor-faktor lainnya (Wulandari et al., 2020).

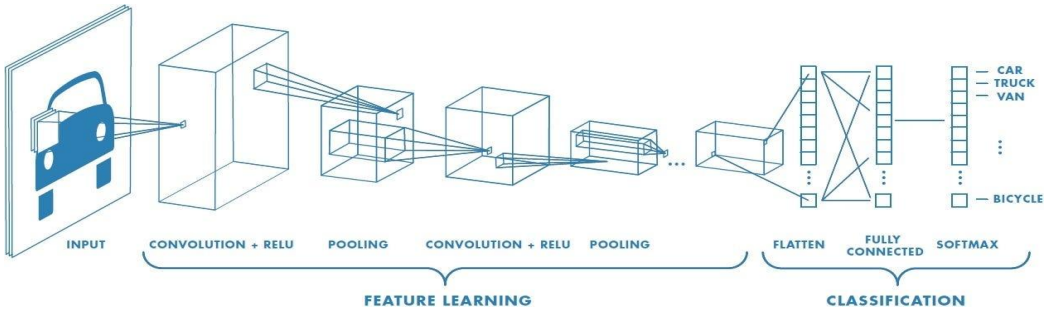
1.6.2 *Deep Learning*

Deep Learning merupakan cabang dari kecerdasan buatan *(Artificial Intelligence)* dan *machine learning* yang dikembangkan dari neural network berlapis untuk meningkatkan *accuracy* dalam berbagai tugas, seperti deteksi objek, pengenalan suara, penerjemahan bahasa, dan aplikasi lainnya. *Deep Learning* bekerja dengan memanfaatkan artificial neural network yang terdiri atas banyak lapisan (*multi-layer*). Jaringan saraf tiruan ini dirancang menyerupai cara kerja otak manusia, di mana neuron-neuron saling terhubung dan membentuk suatu jaringan yang kompleks (Raup et al., 2022).

Deep Learning juga didefinisikan sebagai pendekatan pembelajaran yang menggunakan jaringan saraf buatan yang tersusun dalam beberapa lapisan untuk mempelajari representasi data secara hierarkis. Jaringan saraf buatan tersebut meniru mekanisme kerja otak manusia melalui keterhubungan antarneuron yang membentuk jaringan yang kompleks. *Deep Learning*, yang dikenal pula sebag*ai deep structured learning, hierarchical learning, atau deep neural network*, merupakan metode pembelajaran yang memanfaatkan serangkaian transformasi nonlinier untuk mengekstraksi fitur dari data. Pendekatan ini dapat dipandang sebagai integrasi antara *machine learning* dan kecerdasan buatan yang berbasis jaringan saraf buatan (Nugroho et al., 2020)

1.6.3 *Convolutional Neural Network (CNN)*

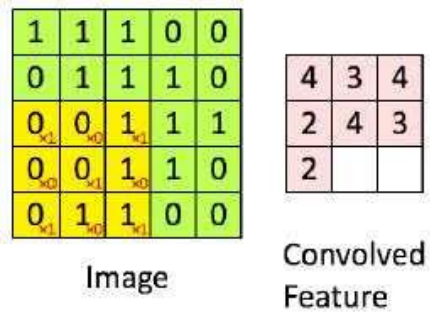
Convolutional Neural Network (CNN) adalah penyempurnaan dari *Multilayer Perceptron (MLP)* yang termasuk dalam kategori jaringan saraf dengan tipe *feed forward*, yang berarti tidak bersifat berulang. *Convolutional Neural Network* dirancang khusus untuk mengolah data dalam format dua dimensi. *CNN* termasuk dalam kelompok *Deep Neural Network* karena memiliki banyak lapisan dan sering digunakan untuk data gambar. *CNN* berfungsi untuk menganalisis gambar visual, mendeteksi, dan mengenali objek dalam sebuah gambar, yang merupakan vektor dengan dimensi tinggi dan melibatkan banyak *parameter* untuk menggambarkan jaringan. Secara umum, *CNN* tidak berbeda jauh dari jaringan saraf konvensional. *CNN* terdiri dari *neuron* yang memiliki bobot, bias, dan fungsi aktivasi (Nugroho et al., 2020). Arsitektur umum Convolutional Neural Network (CNN) ditunjukkan pada Gambar 1.



Gambar 1. Arsitektur *Convolutional Neural Network (CNN)* (Nugroho et al., 2020)

1.6.3.1 *Convolutional Layer*

Convolutional Layer terdiri dari neuron yang diatur sedemikian rupa sehingga membentuk filter dengan ukuran panjang dan tinggi (piksel). Proses konvolusi melibatkan penggunaan *kernel* dan *stride*, di mana proses ini adalah kombinasi dari dua matriks yang berbeda untuk menghasilkan matriks baru. Dalam pemrosesan citra, konvolusi berarti menerapkan sebuah kernel (kotak berwarna kuning) pada citra di semua posisi yang mungkin, seperti yang terlihat dalam ilustrasi pada *Gambar 2* (Azmi et al., 2023).



Gambar 2. Gambaran Operasi Konvolusi (Azmi et al., 2023)

Berdasarkan Gambar 2 kotak berwarna hijau secara keseluruhan merupakan gambar yang akan melalui proses konvolusi. Kernel berpindah dari pojok kiri atas menuju pojok kanan bawah. Dengan demikian, hasil dari konvolusi yang dilakukan pada Gambar 2 bisa dilihat pada gambar di sebelah kanan. Tujuan dari melakukan konvolusi pada data gambar adalah untuk mengambil fitur dari gambar yang diberikan (Azmi et al., 2023).

1.6.3.2 Activation Function (ReLU dan Softmax)

Fungsi aktivasi merupakan sistem yang menentukan hasil dari sebuah *neuron*, baik itu hubungan yang sederhana maupun yang rumit. Fungsi aktivasi diperlukan agar jaringan saraf bisa memahami pola-pola yang lebih rumit dalam data. Beberapa tipe fungsi aktivasi yang sering dipakai meliputi *ReLU (Rectified Linear Unit)* dan *Softmax*. Beberapa fungsi aktivasi yang dipakai dalam penelitian ini antara lain (Purwitasari & Soleh, 2022):

a. Fungsi *ReLU (Rectified Linear Unit)* adalah jenis fungsi aktivasi yang memiliki perhitungan yang mudah. Apabila ada elemen yang bernilai negatif, maka nilai tersebut akan diganti menjadi 0, dan tidak ada tindakan seperti eksponensial, perkalian, atau pembagian. Fungsi aktivasi *ReLU* dinyatakan dalam rumus pada persamaan (1) yaitu:

$$f(x) = \max(0, x) \ atau \qquad (1)$$
$$f(x) = \begin{cases} 0 \ untuk \ x \leq 0 \\ x \ untuk \ x \ > 0 \end{cases}$$

Keterangan:
x = data inputan

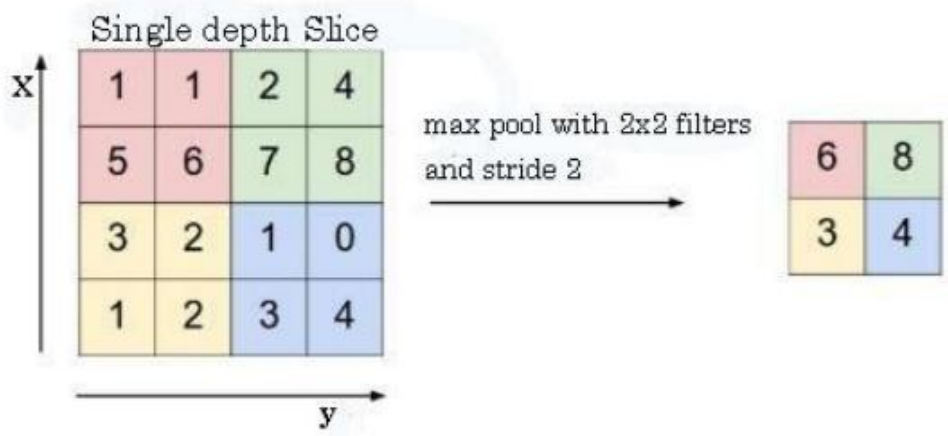
b. Fungsi *Softmax* adalah fungsi yang dipakai untuk menghitung kemungkinan dalam menentukan klasifikasi dengan lebih dari satu kelas, di mana output kelas memiliki nilai kemungkinan tertinggi. Hasil dari fungsi aktivasi *Softmax* memiliki nilai kemungkinan yang berkisar antara 0 hingga 1 (Purwitasari & Soleh, 2022). Rumus fungsi aktivasi *Softmax* ini dijelaskan oleh (Hibatullah & Maliki, 2019) dalam persamaan (2) yaitu:

$$y_i = \frac{e^{y_{in_i}}}{\sum_{i=1}^{m} e^{M}} \quad (2)$$

Keterangan:
y_i = keluaran untuk output layer ke-i
y_{in_i} = masukkan untuk node layer ke-i
M = semua masukkan untuk output layer sejumlah M buah

1.6.3.3 *Pooling Layer*

Lapisan *pooling* berfungsi pada setiap lapisan dari *feature* map dan mengurangi dimensinya. Biasanya, lapisan *pooling* menggunakan filter berukuran 2x2 yang diterapkan dengan langkah dua dan bekerja pada setiap potongan dari inputnya. Di bawah ini ada contoh gambar dari operasi *Max Pooling*:

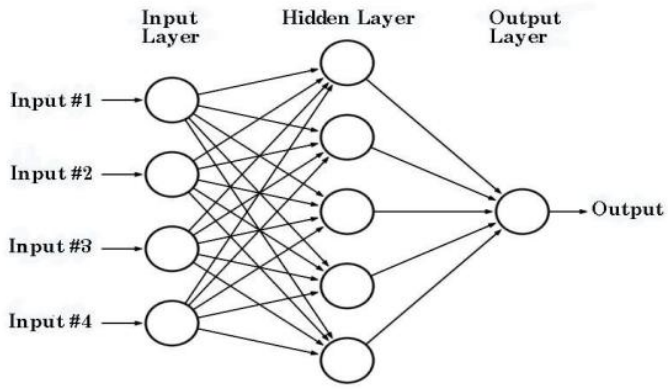


Gambar 3. *Max Pooling* (Azmi et al., 2023)

Berdasarkan Gambar 3 kotak yang memiliki warna merah, hijau, kuning, dan biru di sebelah kiri merupakan kelompok kotak yang akan dipilih untuk mendapatkan nilai tertinggi. Dengan demikian, hasil dari proses ini akan terlihat pada kumpulan kotak yang ada di sebelah kanan (Azmi et al., 2023).

1.6.3.4 *Fully Connected Layer*

Lapisan yang disebut *Fully connected* atau sering disebut dengan *Dense layer* adalah bagian di mana setiap neuron yang bekerja dari lapisan sebelumnya terhubung secara keseluruhan dengan neuron-neuron di lapisan berikutnya, mirip dengan cara kerja jaringan saraf buatan (Azmi et al., 2023). Berikut visualisasi *fully connected layer* pada Gambar 4.



Gambar 4. Visualisasi *Fully Connected Layer* (Azmi et al., 2023)

Fully Connected Layer terdiri dari *input layer*, *hidden layer* dan *output layer* (Hibatullah & Maliki, 2019)

1. *Input layer*
   Input Layer merupakan penggabungan keseluruhan matriks *feature map* yang didapat dari proses *pooling layer* lalu semua piksel tersebut direntangkan menjadi sebuah vektor sepanjang jumlah piksel dari matriks yang didapat pada *pooling layer*. Kemudian seluruh nilai pada *input layer* digunakan untuk perhitungan pada hidden layer.
2. *Hidden Layer*
   Perhitungan pada *layer* ini yaitu mengkalikan nilai-nilai yang ada di input layer dengan bobot yang sudah diinisialisasi sebelumnya lalu ditambahkan dengan nilai bias. Adapun rumus perhitungannya adalah sebagai berikut.

$$z_{in_i} = \sum_{j=1}^{n} X_j * V_{j,i} + V_{0,i} \quad (3)$$

Keterangan:

z_{in_i} = masukkan untuk node hidden layer ke-i dengan jumlah node n
X_j = node X ke-j
$V_{j,i}$ = bobot V untuk X_j dan node Z_i
V_0 = bias V untuk z_{in_i}

3. *Output Layer*
   Perhitungan pada layer ini yaitu mengkalikan nilai-nilai hasil perhitungan pada *hidden layer* dengan bobot yang sudah diinisialisasi sebelumnya lalu ditambahkan dengan nilai bias. Adapun rumus perhitungannya adalah sebagai berikut.

$$y_{in_i} = \sum_{j=1}^{m} Z_j * W_{j,i} + W_{0,i} \quad (4)$$

Keterangan:

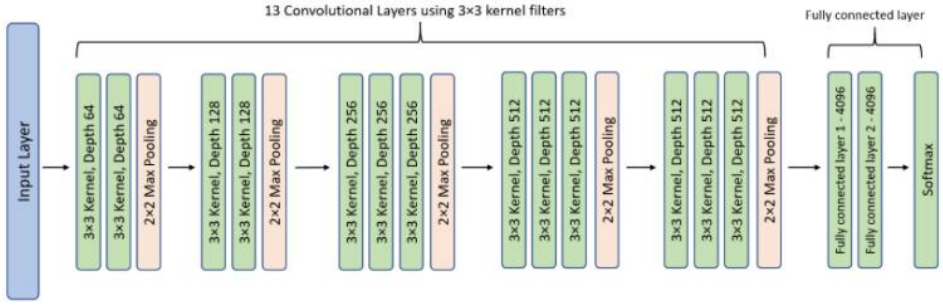
y_{in_i} = masukkan untuk node hidden layer Z ke-I dengan jumlah node m
Z_j = node Z ke-j
$W_{j,i}$ = bobot W untuk Z_j dan node Y_i
$W_{0,i}$ = bias W untuk y_{in_i}

1.6.4 *Transfer Learning*

Transfer Learning merupakan cara atau teknik yang memanfaatkan model yang sudah dilath sebelumnya dan menggunakannya sebagai dasar untuk mempelajari tugas atau permasalahan yang baru. Dengan menggunakan *Transfer Learning*, kemampuan yang sudah dipelajari dapat diterapkan pada pekerjaan baru, sehingga mempercepat proses pembljaran model. Karena model ini sudah melalui pelatihan sebelumnya dan telah belajar untuk mengambil berbagai fitur yang berbeda, model yang baru menjadi lebih tepat dalam menjalankan tugasnya (Faturrahman Et Al., 2023).

1.6.5 *VGG16 (Visual Geometry Group 16)*

VGG16 (Visual Geometry Group 16) merupakan arsitektur *convolutional neural network* dalam yang terdiri dari 16 lapisan. Arsitektur *VGG16* dapat dilihat pada Gambar 5. *VGG16* memiliki 13 *convolutional layer*, 2 *fully connected layer*, dan 1 *classification layer (Rismiyati & Luthfiarta, 2021)*.

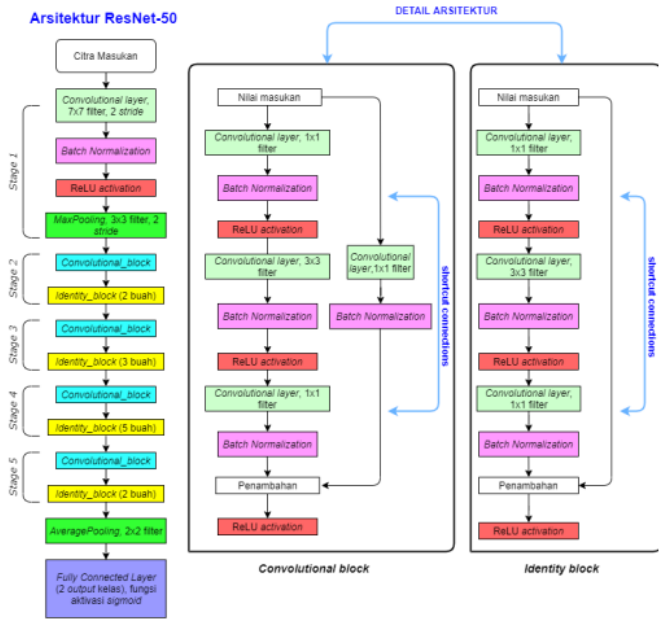


Gambar 5. *Arsitektur VGG16* (Rismiyati & Luthfiarta, 2021)

Pada Gambar 5, setiap *convolutional layer* menggunakan ukuran kernel 3x3. Perbedaan utama antara setiap *convolutional layer* terdapat pada jumlah filter yang ada di masing-masing lapisan. Dua *convolutional layer* yang pertama memiliki 64 filter. Sementara itu, lapisan 3 dan 4 memiliki 128 filter. *Convolutional layer* lainnya juga memiliki jumlah filter yang bervariasi, yaitu 256 pada lapisan 4, 5, dan 6, serta 512 pada lapisan 7, 8, 9, 10, 11, dan 12. Proses *Max Pooling* 2x2 dilakukan setelah *convolutional layer* 2, 4, 7, 10, dan 13. Hasil dari *pooling* terakhir dihubungkan ke *Fully Connected Layer*, dan akhirnya akan dihubungkan ke pengklasifikasi untuk menentukan kategori dari gambar. *(Rismiyati & Luthfiarta, 2021)*

1.6.6 *ResNet50 (Residual Network 50)*

ResNet50 (Residual Network 50) adalah salah satu jenis arsitektur *CNN* yang mengenalkan ide baru yang disebut dengan hubungan cepat atau *shortcut connections.* Kehadiran konsep *shortcut connections* dalam arsitektur *ResNet50* berkaitan dengan masalah *vanishing gradient* yang muncul ketika kita mencoba untuk membuat network lebih dalam. Namun, mencoba untuk memperdalam network demi meningkatkan kinerjanya tidak bisa hanya dilakukan dengan cara menumpuk layer. Semakin dalam suatu *network* dapat menyebabkan masalah *vanishing gradient* yang membuat *gradient* menjadi sangat kecil, sehingga dapat mengurangi kinerja atau *accuracy.* Secara keseluruhan *ResNet50* terdiri dari 5 stage proses konvolusi yangkemudian dilanjutkan *average pooling* dan diakhiri dengan *fully connected layer* sebagai *layer* prediksi. Gambar 6 dibawah mengilustrasikan arsitektur *ResNet50 (Faiz Nashrullah et al., 2020)*



Gambar 6. Diagram blok arsitektur *ResNet50* (Faiz Nashrullah et al., 2020)

1.6.7 *Augmentasi*

Augmentasi merupakan metode yang diterapkan untuk secara artifisial memperbanyak dan memperkaya variasi data pelatihan dengan menerapkan perubahan terhadap data yang sudah ada, seperti memutar, menggeser, memotong, atau membalik. Sasaran dari pendekatan ini adalah untuk mendukung model dalam proses pembelajaran, meningkatkan tingkat generalisasi, serta mengurangi kemungkinan terjadinya *overfitting* (Muttaqin & Widodo, 2025).

1.6.8 *Grad-CAM (Gradient-weighted Class Activation Mapping)*

Grad-CAM adalah metode yang bermanfaat untuk menunjukkan bagian mana dari model *CNN* yang diperhatikan. Metode ini bisa digunakan untuk menjelaskan hasil dari sebuah model pembelajaran mesin. *Grad-CAM* memanfaatkan informasi gradien dari target yang ada di *convolutional layer* terakhir model untuk menciptakan peta lokasi yang menonjolkan area penting dari gambar input untuk prediksi. Selain mampu menjelaskan aktivasi di *convolutional layer* terakhir, metode ini juga bisa diterapkan pada semua *convolutional layer* sehingga konsep keterjelasan dapat dilakukan secara menyeluruh (Muhammad et al., 2023).

1.6.9 *Streamlit*

Streamlit merupakan pustaka *Python* yang bersifat *open-source*, yang memungkinkan para pengembang untuk dengan cepat menciptakan aplikasi web yang menarik dan interaktif dengan sedikit pengkodean. Keunggulan utama dari *Streamlit* adalah kesederhanaan dan efektivitasnya dalam membangun aplikasi yang berfokus pada data. Proses pengembangan *website* konvensional, terutama dalam hal visualisasi data dan aplikasi pembelajaran mesin (Akkem et al., 2023).

1.7 Penelitian Terdahulu

Dalam penyusunan penelitian ini, penulis menggunakan acuan dari berbagai hasil penelitian terdahulu sebagai referensi. Berikut ini penelitian terdahulu yang memiliki keterkaitan dengan topik penelitian ini, antara lain:

Penelitian berjudul “Klasifikasi Isyarat Bahasa Indonesia Menggunakan Metode *Convolutional Neural Network*” yang dilakukan oleh (Al Rivan & Hartoyo, 2022) membahas penerapan arsitektur *CNN VGG16* dan *AlexNet* untuk mengidentifikasi huruf dalam Sistem Isyarat Bahasa Indonesia (SIBI). *Dataset* yang dipakai meliputi huruf dari A sampai Z dengan total 1600 data untuk pelatihan, 320 untuk *validasi*, dan 320 untuk pengujian, yang telah melalui proses *resize, grayscale*, dan augmentasi. Hasil penelitian menunjukkan bahwa arsitektur *VGG16* dengan optimasi Adam memberikan performa

terbaik dengan *accuracy* 99,32% untuk setiap huruf dan 91,18% secara keseluruhan. Sedangkan hasil klasifikasi menggunakan *VGG16* dengan optimasi *SGD* mendapatkan hasil *accuracy* terendah yaitu 98,85% untuk setiap huruf dan 84,96% secara keseluruhan. Di sisi lain, *AlexNet* dengan optimasi *Adam* memperoleh *Accuracy* 99,16% per huruf dan 89,04% secara keseluruhan. Sedangkan hasil klasifikasi menggunakan *AlexNet* dengan optimasi *SGD* mendapatkan hasil *accuracy* terendah yaitu 97,33% untuk setiap huruf dan 68,33% secara keseluruhan. Penelitian ini menegaskan bahwa arsitektur yang lebih dalam seperti *VGG16* mampu menghasilkan klasifikasi yang lebih baik dibandingkan *AlexNet*, serta menunjukkan bahwa optimasi Adam memberikan hasil yang lebih stabil dibandingkan SGD.

Penelitian berjudul "*Comparison of Transfer Learning Algorithm Performance in Hand Sign Language Digits Image Classification*" yang dilakukan oleh (Siddik, 2023). Dalam penelitian ini, dua arsitektur *Transfer Learning* yang dibandingkan adalah *VGG16* dan *ResNet50*, menggunakan *dataset Hand Sign Language Digits* yang terdiri dari 2.062 citra tangan dengan label angka 0 hingga 9. Semua citra diubah ukurannya menjadi 224×224 piksel, kemudian dibagi menjadi data pelatihan dan data pengujian dengan rasio 75:25. Hasil dari pengujian menunjukkan bahwa *VGG16* memberikan hasil terbaik dengan *Accuracy* 97,29%, *presisi* 97,38%, *Recall* 97,45%, dan *F1-Score* 97,36%, sementara *ResNet50* mendapatkan *Accuracy* 94,57%, *presisi* 94,75%, *Recall* 94,96%, dan *F1-Score* 94,78%. Penelitian ini menunjukkan bahwa pemilihan arsitektur *Transfer Learning* berpengaruh besar terhadap hasil klasifikasi, dan pada *dataset* ini, *VGG16* lebih baik daripada *ResNet50*.

Penelitian berjudul "*Deep Transfer Learning for Sign Languange Image Classification: A Bisindo Dataset Study*" yang dilakukan oleh (Rachmawati et al., 2023) meneliti bagaimana *deep Transfer Learning* dalam dapat digunakan untuk mengklasifikasikan gambar BISINDO. Dalam studi ini, beberapa model yang sudah dilatih sebelumnya seperti *ResNet50*, *MobileNetV4, EfficientNetB1, dan InceptionV3* digunakan. Model-model ini kemudian dilatih menggunakan *dataset* BISINDO yang mencakup 26 huruf alfabet beserta variasi latar belakang. Berdasarkan hasil dari penelitian ini menunjukkan bahwa *ResNet50* dan *MobileNetV4* memberikan hasil terbaik dengan tingkat *Accuracy* yang sangat tinggi dalam pelatihan, *validasi*, dan pengujian. Hasil ini menunjukkan bahwa *Transfer Learning* dapat meningkatkan kinerja proses klasifikasi gambar bahasa isyarat, terutama pada *dataset* yang terbatas.

Penelitian berjudul "Perbandingan *Pre-trained CNN*: Klasifikasi Pengenalan Bahasa Isyarat Huruf Hijaiyah" yang dilakukan oleh (Brianorman & Munir, 2023) membandingkan performa empat jenis arsitektur *CNN* yang telah dilatih sebelumnya, yaitu *MnetV2, VGG16, ResNet50, dan Xception*, untuk tugas mengenali huruf hijaiyah. *Dataset* yang dipakai meliputi 6.200 gambar dengan 31 huruf hijaiyah, dan hasil analisis menunjukkan bahwa *MnetV2*, *VGG16*, dan *Xception* berhasil mencapai *Accuracy* 99,85% masing-masing dalam 2, 3, dan 11 *epoch*. Sedangkan *ResNet50* tidak dapat mencapai batas *Accuracy* setelah memproses 100 *epoch* dengan *Accuracy* 82,12%. Untuk proses pengujian menggunakan data uji menunjukkan bahwa *MnetV2*, *Xception*, *VGG16* mendapatkan hasil 100% untuk *precision, Recall, F1-Score, dan accuracy*. Sedangkan, pada pengujian *ResNet50* dihasilkan 81,55%, 86,04%, 82,04%, 82,58% untuk *precision, Recall, F1-Score, dan accuracy* secara

berurutan.Penelitian ini juga berhasil menerapkan model terbaik dalam waktu nyata dengan hasil deteksi yang cepat, sehingga menunjukkan bahwa pilihan arsitektur *CNN* yang tepat sangat mempengaruhi performa dan penerapan aplikasi pengenalan bahasa isyarat secara langsung.

Penelitian berjudul "Deteksi Bahasa Isyarat Berdasarkan SIBI Menggunakan *Transfer Learning*" yang dilakukan oleh (Subkhi et al., 2024) menciptakan sistem untuk mengklasifikasikan huruf dalam bahasa isyarat SIBI dengan menggunakan teknik *Transfer Learning*. Dalam studi ini, terdapat dua jenis model yang digunakan, yaitu *VGG16* dan *MobileNet*, serta dua cara preprocessing yang berbeda: hanya resize dan kombinasi dari *resize*, *flip*, dan *rotate*. Hasil dari pengujian menunjukkan bahwa *MobileNet* memberikan hasil terbaik dengan *Accuracy* mencapai 98%, meskipun waktu komputasinya lebih lama dibandingkan dengan *VGG16* yang hanya mendapatkan *Accuracy* 86%. Penelitian ini mendapatkan hasil terbaik dengan menggunakan menerapkan augmentasi hanya dengan *resize*, serta penelitian ini masih belum membandingkan dengan beberapa *Transfer Learning* lainnya.

Penelitian dengan judul "Penggunaan Metode *Deep Learning* untuk Pengembangan Sistem Komunikasi Cerdas bagi Penyandang Disabilitas" yang dilakukan oleh (Ningsih et al., 2024) bertujuan untuk menciptakan sistem klasifikasi Bahasa Isyarat Indonesia (SIBI) yang menggunakan teknologi *Deep Learning* dengan arsitektur *CNN VGG16* dan VGG19, diintegrasikan ke dalam aplikasi *webiste* untuk mendukung komunikasi bagi penyandang disabilitas, khususnya bagi mereka yang mengalami gangguan pendengaran. *Dataset* SIBI yang digunakan terdiri dari 5.280 gambar yang terbagi dalam 24 kelas yang merepsentasikan huruf A sampai Y, yang telah melalui proses pra-pemrosesan seperti pengubahan ukuran, pengubahan warna, dan berbagai teknik peningkatan data. Model ini dilatih dengan menggunakan *Transfer Learning* dan diatur dengan *parameter* seperti 50 *epoch*, ukuran batch 64, dan algoritma optimasi Adam. Hasil studi ini menunjukkan bahwa model tersebut berhasil mencapai *Accuracy* tinggi sebesar 96,4% dengan nilai kehilangan yang rendah yaitu 0,1055, tanpa tanda-tanda *overfitting*. Penilaian menggunakan *confusion matrix* menunjukan hasil prediksi yang sangat tepat untuk semua kategori, didukung oleh nilai *precision, Recall, dan F1-Score* yang besar. Sistem ini dapat melakukan klasifikasi gerakan secara langsung melalui input gambar atau *webcam*, sehingga berpotensi menjadi alat komunikasi yang berguna bagi penyandang disabilitas.

Penelitian yang berjudul "*Transfer Learning* Model *Evaluation on CNN Algorithm Indonesian Sign Language System* (SIBI)" yang dilakukan oleh (Jollyta et al., 2025) mengulas tentang penilaian enam model *Transfer Learning* yaitu *VGG16*, *VGG19*, *ResNet50*, *Densenet121, Inception-V3, dan MobileNet-V2* dalam mengenali 24 huruf dari Sistem Bahasa Isyarat Indonesia (SIBI). Data yang digunakan terdiri dari 600 gambar huruf SIBI yang diolah dengan TensorFlow melalui langkah-langkah augmentasi, pembagian data 80:20, pelatihan *CNN*, dan evaluasi menggunakan *confusion matrix*. Temuan dari penelitian ini menunjukkan bahwa *MobileNet-V2* adalah model yang paling baik, dengan *Accuracy* 78% pada *epoch* ke-20, sedangkan *ResNet50* memiliki *Accuracy* paling rendah dan sering mengalami kesulitan dalam mengenali sebagian besar huruf. Penelitian ini menekankan bahwa efektivitas model *Transfer Learning* sangat tergantung

pada kualitas data, proses augmentasi, dan kesesuaian arsitektur model dengan karakteristik gambar SIBI.

Penelitian berjudul "*A Comparison Analysis Between ResNet50 and XCeption for Handwritten Hangeul Character using Transfer Learning*" yang dilakukan oleh (Kurniadi et al., 2025) membandingkan kinerja dua arsitektur *CNN* mendalam, yaitu *ResNet50* dan *Xception*, dalam mengklasifikasikan 1.920 gambar huruf Hangeul yang ditulis tangan, yang terbagi dalam 24 kategori. Dengan menggunakan *Transfer Learning*, penambahan data, serta evaluasi melalui *5-fold cross validation*. *ResNet50* mencapai nilai sempurna 100% pada beberapa metrik seperti *Accuracy*, presisi, *Recall*, dan *F1-Score*. Sedangkan *Xception Accuracy* sebesar 99,74% dan *AUC* sebesar 100%. Analisis *Grad-CAM* dimanfaatkan untuk menjelaskan area yang menjadi fokus model dan mengidentifikasi kemungkinan kesalahan pada karakter yang memiliki kemiripan bentuk. Hasil penelitian menunjukkan bahwa *ResNet50* secara konsisten lebih baik daripada *Xception*, meskipun perbedaan keduanya tidak signifikan secara statistik berdasarkan *uji sign test*.

Untuk memberikan gambaran yang lebih jelas, berikut disajikan tabel ringkasan penelitian terdahulu yang relevan dengan topik penelitian ini, yang dapat dilihat pada Tabel 1.

Tabel 1. Ringkasan penelitian terdahulu

No	Tahun	Penulis	Judul	Identifikasi Masalah	Teknik/Metode	Hasil
1	2022	(Al Rivan & Hartoyo, 2022)	Klasifikasi Isyarat Bahasa Indonesia Menggunakan Metode *Convolutional Neural Network*	Penelitian ini membandingkan *VGG16* dan *AlexNet* serta pengaruh *optimizer* untuk klasifikasi huruf SIBI A–Z dengan *Accuracy* tinggi.	*CNN VGG16* dan *AlexNet* diuji pada 1.600 data latih, 320 *validasi*, 320 uji, dengan preprocessing dan optimzer Adam/SGD, dievaluasi lewat *Accuracy* per huruf dan keseluruhan.	*VGG16 Adam* 99,32% / 91,18%, *AlexNet* Adam 99,16% / 89,04%, SGD terendah, terutama *AlexNet* 68,33%.
2	2023	(Siddik, 2023)	Comparison of *Transfer Learning* Algorithm Performance in	Menentukan arsitektur *Transfer Learning* terbaik untuk klasifikasi	Diuji dua arsitektur *Transfer Learning*, *VGG16* dan *ResNet50*, pada *dataset* 2.062	*VGG16* unggul dengan *Accuracy* 97,29% dibanding *ResNet50*

No	Tahun	Penulis	Judul	Identifikasi Masalah	Teknik/Metode	Hasil
			Hand Sign Language Digits Image Classification	digit isyarat pada *dataset* kecil.	gambar Sign Language Digits 224×224 px (75% latih, 25% uji), dievaluasi dengan *Accuracy*, presisi, *Recall*, dan *F1-Score*.	94,57%, serta metrik lain lebih tinggi, menunjukkan *VGG16* paling cocok untuk *dataset* ini.
3	2023	(Rachmawati et al., 2023)	Deep *Transfer Learning* for Sign Language Image Classification: A Bisindo *Dataset* Study	Mengkategorikan *dataset* BISINDO dan membandingkan performa beberapa model *Transfer Learning* untuk klasifikasi citra.	Diuji beberapa model *Transfer Learning* (*ResNet50*, MobileNetV4, EfficientNetB1, InceptionV3) pada *dataset* BISINDO 26 huruf, dengan fine-tuning dan evaluasi pada data latih, *validasi*, dan uji.	*ResNet50* dan MobileNetV4 paling stabil dengan *Accuracy* tinggi di semua tahap, membuktikan *Transfer Learning* efektif untuk *dataset* BISINDO yang beragam.
4	2023	(Brianorman & Munir, 2023)	Perbandingan *Pre-trained CNN*: Klasifikasi Pengenalan Bahasa Isyarat Huruf Hijaiyah	Kebutuhan mengembangkan model pengenalan gestur jari untuk huruf hijaiyah sebagai sarana membaca Al-Quran bagi kaum tuli.	Membandingkan 4 model *Pre-trained CNN* (MnetV2, *VGG16*, *ResNet50*, Xception) untuk klasifikasi citra gestur huruf hijaiyah.	Model MnetV2, *VGG16*, dan Xception mencapai *Accuracy* 99,85%, sedangkan *ResNet50* 82,12%

No	Tahun	Penulis	Judul	Identifikasi Masalah	Teknik/Metode	Hasil
5	2024	(Subkhi et al., 2024)	Deteksi Bahasa Isyarat Berdasarkan SIBI (Sistem Bahasa Isyarat) menggunakan *Transfer Learning*	Perbandingan efisiensi dan *Accuracy* *VGG16* dan MobileNet serta pengaruh preprocessing (resize vs. resize+augmentasi).	Diuji *VGG16* dan MobileNet dengan dua preprocessing: (1) resize, (2) resize + flip + rotate, dievaluasi lewat *Accuracy* dan waktu komputasi.	MobileNet *Accuracy* 98% tapi lebih lambat, *VGG16* 86% lebih cepat; hasil terbaik dengan preprocessing resize saja.
6	2024	(Ningsih et al., 2024)	Penggunaan Metode *Deep Learning* untuk Pengembangan Sistem Komunikasi Cerdas bagi penyandang disabilitas	Kurangnya platform online yang mudah diakses dan user-friendly untuk demonstrasi dan pengujian model klasifikasi bahasa isyarat.	Diuji *CNN* *VGG16* dan VGG19 dengan *Transfer Learning* pada 5.280 gambar SIBI (24 kelas), preprocessing resize, perubahan warna, dan augmentasi; dilatih 50 *epoch*, *batch size* 64, *optimizer* Adam, dievaluasi dengan loss, *Accuracy*, dan confusion matrix.	*VGG16* mencapai *Accuracy* 96,4% dengan loss 0,1055, stabil tanpa *overfitting*, dan mampu klasifikasi *realtime* via webcam dengan prediksi optimal.
7	2025	(Jollyta et al., 2025)	*Transfer Learning* Model Evaluation on *CNN* Algorithm Indonesian Sign Language	Evaluasi diperlukan untuk membandingkan kinerja berbagai model *Transfer Learning*	Diuji enam model *Transfer Learning* (*VGG16*, VGG19, *ResNet50*, *Dense*Net121, Inception-V3, MobileNetV2) pada 600	MobileNetV2 menunjukkan performa terbaik dengan *Accuracy* 78%, sedangkan

No	Tahun	Penulis	Judul	Identifikasi Masalah	Teknik/Metode	Hasil
			System (SIBI)	pada *dataset* SIBI kecil dan perbedaan kemampuan *CNN* mengenali alfabet SIBI.	gambar SIBI (24 alfabet) dengan augmentasi, train-test split 80:20, pelatihan *CNN*, dan evaluasi menggunakan confusion matrix.	*VGG16*, VGG19, Inception-V3, *Dense*Net121, dan terutama *ResNet50* memiliki *Accuracy* lebih rendah.
8	2025	(Kurniadi et al., 2025)	A Comparison Analysis Between *ResNet50* and XCeption for Handwritten Hangeul Character using *Transfer Learning*	Kesulitan klasifikasi karakter Hangeul bahasa Korea karena kesamaan antar karakter dan struktur suku kata yang kompleks.	Menggunakan *Transfer Learning ResNet50* dan Xception, ditambah 5-fold cross-*validation*. Visualisasi *Grad-CAM* digunakan untuk mengidentifikasi area fokus model. *Dataset* 1.920 gambar, 24 kelas.	*ResNet50* berkinerja lebih baik, mencapai *Accuracy*, presisi, *Recall*, dan *F1-Score* 100%. Xception mencapai *Accuracy* hingga 99.74%.

BAB II
METODE PENELITIAN

2.1 Pendekatan Penelitian

Penelitian ini menggunakan pendekatan kuantitatif dengan membandingkan model *VGG16* dan *ResNet50* melalui metode *Transfer Learning*. Data citra angka bahasa isyarat diuji menggunakan berbagai jenis *dataset*, yaitu *dataset* asli, *dataset* yang telah diaugmentasi, dan *dataset* gabungan, dengan skema pelatihan secara *independen* serta fine-tuning. Proses penelitian mencakup tahap preprocesing, pelatihan, dan evaluasi model guna memperoleh model yang terbaik. Untuk meningkatkan transparansi sistem, metode *Grad-CAM* diterapkan sebagai bagian dari *Explainable AI* untuk memvisualisasikan dasar pengambilan keputusan oleh model. Hasil penelitian kemudian diintegrasikan ke dalam *Dashboard Streamlit* sebagai solusi komunikasi inklusif yang interaktif dan berjalan secara *real-time*.

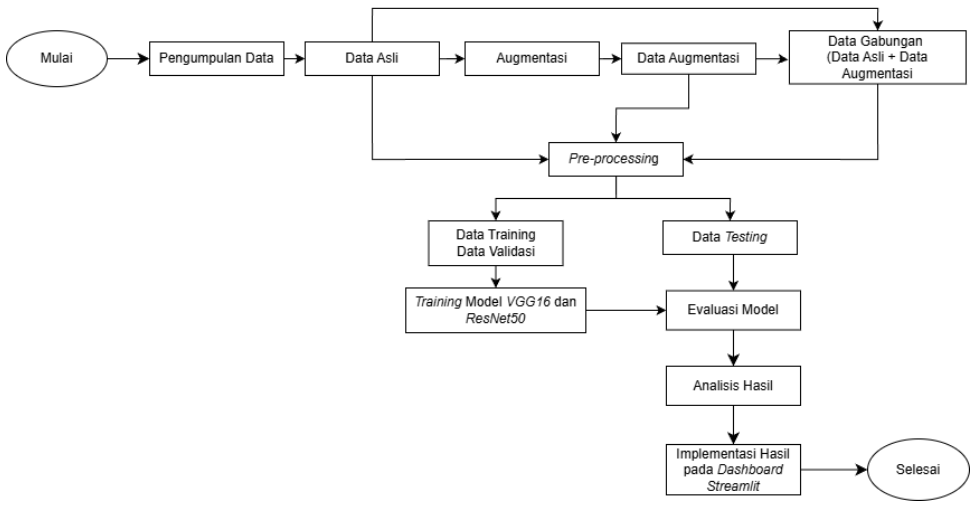
2.2 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan selama kurang lebih 3 bulan, dimulai dari bulan September hingga bulan November 2025. Penelitian ini bertempat di Laboratorium Rekayasa Perangkat Lunak Fakultas MIPA. Adapun untuk detail waktu penelitian dapat dilihat pada Tabel 2.

Tabel 2. Waktu penelitian

Kegiatan	September				Oktober				November			
	1	2	3	4	1	2	3	4	1	2	3	4
Studi Literatur												
Pengumpulan Data												
Pre-processing Data												
Training Model												
Evaluasi Model												
Analisis Hasil												
Deploy Model												

Untuk memberikan gambaran yang lebih jelas mengenai tahapan yang dilakukan selama proses penelitian, berikut ini adalah alur penelitian yang menjelaskan langkah-langkah mulai dari pengumpulan data hingga evaluasi hasil model, yang dapat dilihat pada Gambar 7.



Gambar 7. Diagram alur penelitian

2.3 Deksripsi Data

Dataset yang digunakan pada penelitian ini merupakan *Sign Language Digits Dataset*, yaitu kumpulan citra tangan manusia yang membentuk angka 0 sampai 9. Kumpulan data ini diambil dari situs Kaggle (https://www.kaggle.com/datasets/ardamavi/sign-language-digits-dataset), sebuah platform yang menyediakan berbagai *dataset* publik dan banyak digunakan oleh peneliti maupun praktisi dalam bidang *machine learning*. *Dataset* ini berisi 2.062 citra dengan resolusi 100 x 100 piksel, dan terbagi ke dalam 10 kelas, yaitu representasi angka 0 sampai 9. Setiap kelas memiliki jumlah citra yang berbeda-beda, namun tetap berada dalam kisaran 204 hingga 208 citra per kelas.

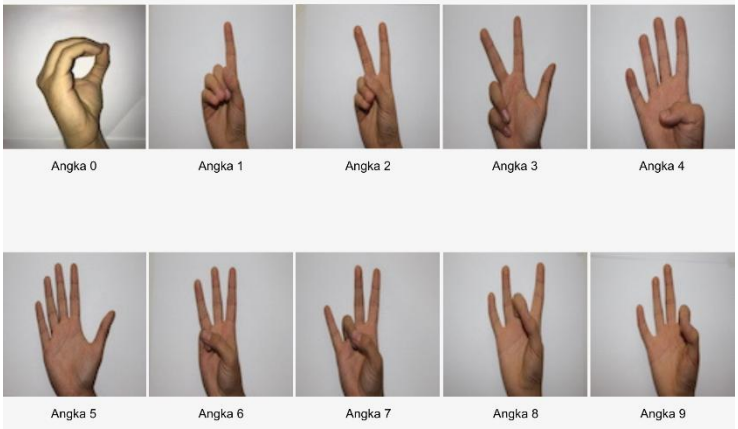
Untuk meningkatkan keragaman data dan memperbaiki kemampuan generalisasi mode, dilakukan augmentasi rotasi pada seluruh citra asli. Proses augmentasi ini menghasilkan 2.062 citra baru, sehingga total keseluruhan *dataset* menjadi 4.124 citra. Augmentasi ini tidak mengubah label kelas, sehingga distribusi jumlah citra perkelas tetap sama. Distribusi lengkap jumlah citra sebelum, sesudah augmentasi dan data gabungan dapat dilihat pada Tabel 3, Tabel 4 dan Tabel 5 yang menunjukkan bahwa setiap kelas memiliki jumlah data yang seimbang.

Tabel 3. Distribusi kelas *dataset* asli

Label	Jumlah
0	205
1	206
2	206
3	206
4	207
5	207
6	207

Label	Jumlah
7	206
8	208
9	204

Contoh citra sebelum di augmentasi rotasi dapat dilihat pada Gambar 8.



Gambar 8. Contoh Citra Sebelum Proses Augmentasi Rotasi

Setelah dilakukan augmentasi rotasi, jumlah citra pada setiap kelas tetap sama, sebagaimana ditunjukkan pada Tabel 4 berikut.

Tabel 4. Distribusi kelas *dataset* hasil augmentasi

Label	Jumlah
0	205
1	206
2	206
3	206
4	207
5	207
6	207
7	206
8	208
9	204

Contoh citra setelah augmentasi rotasi dapat dilihat pada Gambar 9.

Gambar 9. Contoh Citra Setelah Proses Augmentasi Rotasi

Tabel 5. Distribusi kelas *dataset* gabungan

Label	Jumlah
0	410
1	412
2	412
3	412
4	414
5	414
6	414
7	412
8	416
9	408

2.4 Instrumen Penelitian

Dalam pelaksanan penelitian ini terdapat beberapa instrumen dalam membantu proses penelitian agar berjalan secara lancar. Instrumen tersebut terbagi menjadi dua yaitu perangkat keras (*hardware*) dan perangkat lunak (*software*).

2.4.1 Perangkat keras (*hardware*)

Perangkat keras yang digunakan dalam penelitian ini dapat dilihat pada Tabel 6.

Tabel 6. Perangkat keras (*hardware*)

Jenis	Spesifikasi
Sistem Operasi	*Windows 11*
Prosesor	*AMD Ryzen 3 3250U with Radeon Graphics (2.60 GHz)*

Jenis	Spesifikasi
Memori (RAM)	*12 GB*
GPU	*AMD Radeon (TM) Graphics*

2.4.2 Perangkat lunak (*software*)

Terdapat beberapa perangkat lunak atau *software* yang digunakan selama penelitian ini yang bertujuan untuk mempermudah proses pengerjaan. Perangkat lunak yang di gunakan dapat di lihat pada Tabel 7.

Tabel 7. Perangkat lunak (*software*)

Perangkat Lunak	Fungsi Utama
Python	Bahasa pemrograman utama yang digunakan dalam penelitian ini untuk mengembangkan, melatih, serta mengevaluasi model *Deep Learning*
Numpy	*Library Python* untuk komputasi numerik yang efisien, khususnya dalam pengolahan data berbasis array dan matriks yang digunakan sebagai representasi citra.
Pandas	*Library Python* untuk mengolah dan menganalisis *dataset* dengan struktur *DataFrame*.
Matplotlib & Seaborn	*Library* visualisasi data yang digunakan untuk menghasilkan grafik, diagram, serta visualisasi tren seperti *loss, accuracy, dan confusion matrix*.
Scikit-Learn	*Library machine learning* yang digunakan untuk proses pembagian data, encoding label, menghitung metrik evaluasi (*precision, Recall, F1-Score*), dan pembuatan *confusion matrix*.
TensorFlow / Keras	*Framework Deep Learning* yang digunakan untuk membangun, melatih, mem*validasi*, dan mengevaluasi arsitektur model seperti *VGG16* dan *ResNet50*.
Augmentor	*Library Python* yang digunakan untu meningkatkan variasi *dataset* melalui augmentasi rotasi
OpenCV (cv2)	*Library* pengolahan citra untuk membaca gambar, mengubah ukuran, dan mempersiapkan citra sebelum masuk ke model *Deep Learning*.
MediaPipe	*Framework* visi komputer yang digunakan pada *Dashboard* untuk mendeteksi tangan secara *realtime* dan menampilkan titik-titik (hand landmarks) pada area tangan ketika dilakukan pendeteksian.
Google Colab	Platform berbasis cloud yang digunakan untuk menjalankan kode *Python*, melakukan *training*

Perangkat Lunak	Fungsi Utama
	model, serta memanfaatkan *GPU* untuk mempercepat proses komputasi.
Streamlit	*Framework Python* yang digunakan untuk membangun *Dashboard* atau antarmuka *website* interaktif untuk menampilkan hasil visualisasi model
Visual Studio Code	*Framework* editor kode yang digunakan untuk menulis, mengedit, dan mengelola program dalam penelitian.

2.5 Teknik Analisis Data dan Pemodelan

Teknik analisis data dalam penelitian ini bertujuan untuk mengembangkan model klasifikasi bahasa isyarat yang paling efektif dengan menerapkan metode *Transfer Learning* pada arsitektur *VGG16* dan *ResNet50*. Dalam penelitian ini terdapat tiga jenis *dataset* yang digunakan, yaitu *dataset* asli, *dataset* augmentasi, serta *dataset* gabungan dari data asli dan data augmentasi. Ketiga jenis *dataset* tersebut melewati prosedur pengolahan yang sama agar hasil evaluasi performa model dapat dibandingkan secara adil dan objektif. Proses analisis mencakup tahapan augmentasi citra, preprocessing, *resize* data, pembagian data menjadi bagian pelatihan dan pengujian, pembuatan arsitektur model, penentuan nilai *hyperparameter*, serta evaluasi performa model.

2.5.1 Augmentasi Citra

Pada tahap ini proses augmentasi citra dilakukan menggunakan *Library Augmentor* untuk menghasilkan citra baru dari *dataset* asli melalui transformasi rotasi. Setiap kelas pada *dataset* diproses secara terpisah dengan membangun pipeline augmentasi pada folder masing-masing kelas. Pipeline tersebut kemudian dijalankan untuk menghasilkan citra baru dengan jumlah yang setara dengan jumlah citra asli pada kelas yang sama.

Dalam penelitian ini, augmentasi yang digunakan berfokus pada rotasi dengan sudut kemiringan maksimal 25° ke kiri maupun ke kanan. Nilai *probability* sebesar 1.0 menunjukkan bahwa seluruh citra pada kelas tersebut akan dikenai transformasi rotasi. Proses ini menghasilkan 2.062 citra baru, sehingga total keseluruhan *dataset* bertambah menjadi 4.124 citra.

2.5.2 Tahapan *Pre-processing* Data

Setelah data terkumpul, data tersebut akan melalui proses *pre-processing* sebelum dapat dianalisis dan digunakan untuk pemodelan. *Dataset* diproses melalui serangkaian tahapan *pre-processing* yang bertujuan menyiapkan data agar sesuai untuk proses pelatihan model. Tahapan ini dilakukan untuk meningkatkan kualitas data serta memastikan model dapat mengenali pola dengan lebih optimal. Adapun tahapan *pre-*

processing dalam penelitian ini meliputi penyesuaian ukuran citra (resize) dan pembagian data (*splitting data*).

2.5.2.1 *Resize Citra*

Pada tahap *pre-processing* ini, seluruh citra pada *dataset* diubah ukurannya dari 100 × 100 piksel menjadi 224 × 224 piksel. Proses penyesuaian ukuran dilakukan menggunakan *library OpenCV*, yang berfungsi untuk memastikan setiap citra memiliki dimensi yang seragam sebelum dimasukkan ke dalam model. Ukuran 224 × 224 dipilih karena merupakan ukuran input standar untuk arsitektur *VGG16* dan *ResNet50*. Dengan penyesuaian ini, seluruh citra dapat diproses secara konsisten sehingga model mampu melakukan ekstraksi fitur dengan lebih optimal.

2.5.2.2 Pembagian Data *(Splitting data)*

Setelah proses pengubahan ukuran citra (*resize*), tahap selanjutnya adalah melakukan pembagian data (*splitting data*) untuk keperluan pelatihan dan evaluasi model. Pada penelitian ini, *dataset* dipisahkan menjadi tiga bagian, yaitu data *training*, data *validasi*, dan data uji. Untuk *dataset* asli dan *dataset* augmentasi yang masing-masing berjumlah 2.062 citra, pembagian awal dilakukan dengan alokasi 80% atau 1.649 citra sebagai data *training* awal dan 20% atau 413 citra sebagai data uji.

Selanjutnya, sebanyak 1.649 citra data *training* awal tersebut dibagi kembali dengan proporsi 80% untuk data *training* dan 20% untuk data *validasi*, sehingga menghasilkan 1.319 citra sebagai data *training* dan 330 citra sebagai data *validasi*. Proses pembagian yang sama juga diterapkan pada *dataset* gabungan yang berjumlah 4.124 citra, di mana 826 citra digunakan sebagai data uji dan 3.298 citra menjadi data *training* awal. Dari data *training* awal tersebut, sebanyak 2.638 citra dialokasikan sebagai data *training* dan 660 citra sebagai data *validasi.* Dengan demikian, setiap jenis *dataset* memiliki komposisi pembagian data yang konsisten, yaitu 64% untuk data *training*, 16% untuk data *validasi*, dan 20% untuk data uji. Berikut Tabel 8 pembagian data pada setiap jenis *dataset*.

Tabel 8. Hasil pembagian data (*splitting data*)

Data	Total data	Data Train 64%	Data *Validation* 16 %	Data Test 20%
Data Asli	2062	1319	330	413
Data Augmentasi	2062	1319	330	413
Data Gabungan	4124	2638	660	826

2.5.3 Arsitektur Model

Dalam penelitian ini, proses pemodelan dilakukan menggunakan Keras dengan *TensorFlow* sebagai *backend*. Dua arsitektur utama yang digunakan adalah *VGG16* (*Visual Geometry Group 16*) dan *ResNet50* (*Residual Network 50*). Kedua model tersebut diterapkan menggunakan pendekatan *Transfer Learning* dan digunakan untuk melakukan klasifikasi citra bahasa isyarat berdasarkan 10 kelas yang telah ditentukan.

2.5.3.1 *VGG16 (Visual Geometry Group 16)*

Pada penelitian ini, *VGG16* digunakan sebagai salah satu arsitektur pre-trained *Convolutional Neural Network (CNN)* untuk melakukan klasifikasi citra bahasa isyarat. Model diimplementasikan menggunakan pendekatan *Transfer Learning* dengan memanfaatkan *convolutional layer VGG16* sebagai pemroses fitur utama. Versi *VGG16* yang digunakan adalah model tanpa lapisan fully connected bawaan pada bagian akhir jaringan (*include_top = False*), sehingga hanya bagian ekstraksi fitur yang dipertahankan.

Pada tahap awal, model menerima masukan berupa citra berukuran 224 × 224 × 3 sesuai dengan standar arsitektur *VGG16*. Seluruh *parameter* pada model dasar kemudian dibekukan (*non-trainable*) agar bobot hasil pelatihan pada *dataset ImageNet* tetap dipertahankan serta untuk meningkatkan efisiensi proses pelatihan. Setelah itu, beberapa lapisan klasifikasi tambahan ditambahkan untuk menyesuaikan arsitektur model dengan kebutuhan klasifikasi 10 kelas pada penelitian ini. Lapisan *GlobalAveragePooling2D* digunakan untuk mereduksi keluaran fitur berukuran 7 × 7 × 512 menjadi vektor satu dimensi sepanjang 512. Selanjutnya, lapisan *Dense* dengan 1.024 unit dan fungsi aktivasi *ReLU* ditambahkan sebagai tahap pemrosesan lanjutan. Pada tahap akhir, lapisan *Dense* dengan 10 unit dan fungsi aktivasi *Softmax* digunakan sebagai lapisan output untuk menghasilkan prediksi kelas akhir.

Tabel 9. Arsitektur model *VGG16*

Model : *"sequeintial"*			
Nama *Layer (type)*	**Fungsi Aktivasi**	***Output Shape***	**Jumlah *Parameter***
VGG16 (functional)	-	(None 7, 7, 512)	14.714.688
Global_average_pooling2d (GlobalAveragePooling2D)	-	(None, 512)	0
Dense (Dense)	*ReLU*	(None, 1024)	525.312
Dense_1 (Dense)	*Softmax*	(None, 10)	10.250
Total params: 16.321.376 (62.26 MB) *Trainable params: 535.562 (2.04 MB)* *Non-trainable params: 14.714.688 (56.13 MB)* *Optimizer params: 1.071.126 (4.09 MB)*			

Berdasarkan Tabel 9, model *VGG16* yang digunakan memiliki total 16.321.376 *parameter*, yang mencakup seluruh bobot dan bias dari arsitektur model. Dari jumlah

tersebut, sebanyak 14.714.688 *parameter* termasuk dalam kategori *non-trainable parameters* karena berasal dari model dasar *VGG16* yang digunakan sebagai feature extractor dan tidak diperbarui selama proses pelatihan. Sementara itu, sebanyak 535.562 *parameter* merupakan trainable *parameters* yang terdapat pada lapisan-lapisan tambahan, seperti lapisan *Dense* pada bagian klasifikasi, sehingga *parameter* inilah yang dilatih untuk menyesuaikan model dengan *dataset* penelitian. Selain *parameter* model, sistem juga mencatat adanya 1.071.126 *parameter* yang berkaitan dengan mekanisme *optimizer*, yang bersifat internal dan digunakan untuk mendukung proses pembaruan bobot selama pelatihan, namun tidak termasuk sebagai bagian dari arsitektur jaringan.

2.5.3.2 *ResNet50 (Residual Network 50)*

Pada penelitian ini, *ResNet50* digunakan sebagai arsitektur pre-trained *Convolutional Neural Network (CNN)* untuk melakukan klasifikasi citra bahasa isyarat. Pemanfaatan *ResNet50* dilakukan dengan pendekatan *Transfer Learning*, di mana model digunakan sebagai feature extractor dengan memanfaatkan bobot hasil pelatihan pada *dataset ImageNet*.

Model *ResNet50* menghilangkan lapisan fully connected bawaan pada bagian akhir jaringan (*include_top = False*), sehingga hanya bagian konvolusi yang digunakan untuk mengekstraksi fitur dari citra masukan. Dengan konfigurasi tersebut, model menerima citra berukuran 224 × 224 × 3 dan menghasilkan peta fitur berukuran 7 × 7 × 2048 pada lapisan akhir bagian konvolusi. Seluruh *parameter* pada model dasar *ResNet50* dibekukan (*non-trainable*) agar informasi fitur yang telah dipelajari dari *ImageNet* tetap dipertahankan selama proses pelatihan. Selanjutnya, ditambahkan beberapa lapisan klasifikasi baru sebagai penyesuaian terhadap kebutuhan penelitian. Lapisan *GlobalAveragePooling2D* digunakan untuk mereduksi peta fitur berukuran 7 × 7 × 2048 menjadi vektor satu dimensi sepanjang 2048. Vektor fitur tersebut kemudian diproses oleh lapisan *Dense* dengan 1.024 unit dan fungsi aktivasi *ReLU* untuk meningkatkan kemampuan representasi fitur. Pada tahap akhir, lapisan *Dense* dengan 10 unit dan fungsi aktivasi *Softmax* digunakan untuk menghasilkan probabilitas prediksi terhadap 10 kelas angka bahasa isyarat yang diteliti.

Tabel 10. Arsitektur model *ResNet50*

Model : *"sequential"*			
Nama *Layer (type)*	**Fungsi Aktivasi**	***Output Shape***	**Jumlah *Parameter***
ResNet50 (functional)	-	*(None, 7, 7, 2048)*	23.587.712
Global_average_pooling2d (GlobalAveragePooling2D)	-	*(None, 2048)*	0
Dense (Dense)	*ReLU*	*(None, 1024)*	2.098.176
Dense_1 (Dense)	*Softmax*	*(None, 10)*	10.250
Total params: 29.912.992 (114.11 MB) *Trainable params: 2.108.426 (8.04 MB)* *Non-trainable params: 23.587,712 (89.98 MB)* *Optimizer params: 4.216.854 (16.09 MB)*			

Pada

Tabel ***10***, model *ResNet50* memiliki total 29.912.992 *parameter*, yang mencakup seluruh bobot dan bias dari arsitektur model yang digunakan, baik pada bagian ekstraksi fitur maupun lapisan klasifikasi tambahan. Dari jumlah tersebut, sebanyak 23.587.712 *parameter* termasuk dalam kategori *non-trainable parameter*s, karena berasal dari model dasar *ResNet50* yang digunakan sebagai feature extractor dan tidak diperbarui selama proses pelatihan. Sementara itu, sebanyak 2.108.426 *parameter* merupakan trainable *parameter*s yang terdapat pada lapisan-lapisan tambahan, seperti lapisan *Dense* pada bagian klasifikasi. *Parameter* inilah yang dilatih untuk menyesuaikan model dengan karakteristik *dataset* angka bahasa isyarat yang digunakan dalam penelitian ini.

Selain *parameter* model, sistem juga mencatat adanya 4.216.854 *parameter* yang berkaitan dengan mekanisme *optimizer*, yang digunakan untuk mendukung proses pembaruan bobot selama pelatihan. *Parameter* ini bersifat internal pada *optimizer* dan tidak termasuk sebagai bagian dari arsitektur jaringan. Secara keseluruhan, meskipun *ResNet50* memiliki kompleksitas arsitektur yang tinggi, proses pelatihan difokuskan pada lapisan akhir yang relatif ringan sehingga model tetap mampu mencapai performa prediksi yang efektif.

2.5.4 *Hyperparameter*

Pada tahap ini, model yang telah dibuat akan menjalani proses pelatihan dan pengujian guna menilai sejauh mana kemampuannya dalam mengklasifikasi bahasa isyarat Proses ini dilakukan setelah data telah disesuaikan dan dibagi menjadi tiga bagian, yaitu data *training*, data *validation*, dan data *testing*. Eksperimen dilakukan dengan menguji berbagai variasi *parameter* hiper, yakni ukuran batch dan jumlah *epoch*. Kombinasi *hyperparameter* yang digunakan dalam penelitian ini dapat dilihat pada Tabel 11.

Tabel 11. Kombinasi *hyperparameter* pelatihan model

Hyperparameter	**Konfigurasi**
Batch size	32
Epoch	50
Optimizer	*Adam*
Learning rate	0.0001
Loss function	*Sparse Categorical Crossentropy*

Setelah kombinasi *hyperparameter* ditentukan, proses pelatihan model dilakukan pada tiga jenis data, yaitu data asli, data yang telah diaugmentasi, dan data gabungan. Pelatihan ini menggunakan arsitektur *VGG16* dan *ResNet50* sebagai metode *Transfer Learning*. Pada tahap ini, model dilatih dengan bobot awal yang telah dilatih sebelumnya dari *ImageNet*, yang berfungsi sebagai fitur awal. Selanjutnya, seluruh proses pembelajaran difokuskan pada penyesuaian *parameter* model melalui data pelatihan, sehingga model dapat mempelajari pola visual dari setiap kelas bahasa isyarat. Selama pelatihan berlangsung, data *validasi* digunakan untuk memantau stabilitas dan konsistensi kinerja model pada data yang tidak digunakan untuk pelatihan.

Dengan demikian, metrik seperti *accuracy* dan loss pada setiap *epoch* dapat diamati untuk menilai apakah model sedang belajar secara efektif atau mengalami kecenderungan *overfitting*. Setelah proses pelatihan selesai, model yang dihasilkan dari setiap jenis data dievaluasi menggunakan data uji guna memperoleh gambaran objektif mengenai kemampuan model dalam melakukan klasifikasi sebelum masuk ke tahap evaluasi lebih lanjut.

2.5.5 Metrik Evaluasi Kinerja Model

Ada beberapa metrik yang digunakan pada penelitian ini diantaranya (Siddik, 2023). Metrik-metrik ini digunakan untuk memberikan penilaian yang lebih lengkap terhadap hasil prediksi model, sehingga kualitas klasifikasi dapat diketahui tidak hanya dari jumlah prediksi yang benar, tetapi juga dari ketepatan dan kemampuan model dalam mengenali kelas positif.

2.5.5.1 *Accuracy*

Accuracy mengukur seberapa baik model klasifikasi dapat mengklasifikasikan data dengan benar secara keseluruhan. *accuracy* dinyatakan dalam persentase dan dihitung dengan membagi jumlah prediksi yang benar dengan total jumlah sampel

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

Keterangan:

TP (True Positive) adalah kasus ketika model secara benar memprediksi sampel sebagai positif (klasifikasi benar), dan label sejati juga positif

TN (True Negative) adalah kasus ketika model secara benar memprediksi sampel sebagai negatif (klasifikasi benar), dan label sejati juga negatif

FP (False Positive) adalah kasus ketika model secara salah memprediksi sampel sebagai positif (klasifikasi salah), tetapi label sejati sebenarnya negatif

FN (False Negative) adalah kasus ketika model secara salah memprediksi sampel sebagai negatif (klasifikasi salah), tetapi label sejati sebenarnya positif

2.5.5.2 *Precission*

Precission menggambarkan sejauh mana hasil positif yang diprediksi adalah benar

$$Precission = \frac{TP}{TP + FP} \quad (6)$$

2.5.5.3 *Recall*

Recall menggambarkan sejauh mana hasil positif yang sebenarnya ditemukan oleh algoritma

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

2.5.5.4 *F1-Score*

F1-Score menggabungkan presisi dan *Recall* dalam satu metrik tunggal. Ini adalah rata-rata harmonik antara presisi dan *Recall*, yang memberikan perhatian seimbang pada kedua metrik tersebut

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{8}$$

2.6 Rancangan *Dashboard Streamlit*

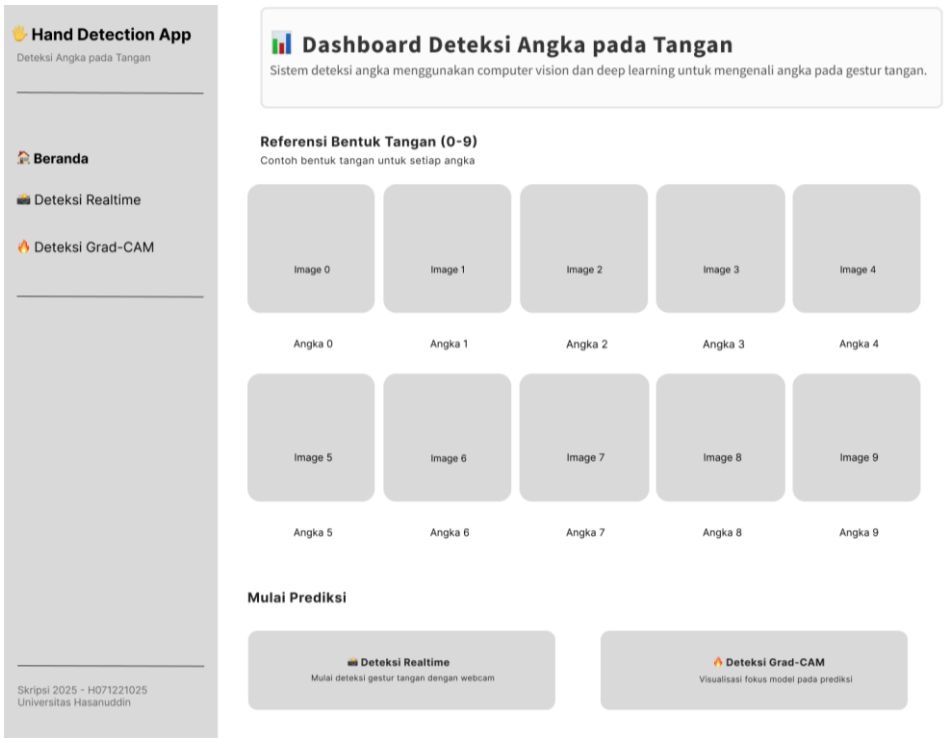
Dalam penelitian ini dikembangkan sebuah antarmuka pengguna atau *Dashboard* menggunakan *Streamlit*, yaitu *Framework open-source* yang memungkinkan pembangunan aplikasi *website* berbasis data *science* secara cepat dan efisien menggunakan bahasa pemrograman *Python*. *Dashboard* yang dihasilkan terdiri dari tiga halaman utama, yaitu halaman beranda, halaman deteksi *realtime*, dan halaman deteksi *Grad-CAM*.

Halaman beranda berfungsi sebagai tampilan awal yang menampilkan judul sistem, deskripsi singkat, referensi bentuk tangan untuk setiap kelas angka, serta ringkasan fitur utama yang dapat diakses oleh pengguna. Halaman deteksi *realtime* menampilkan proses deteksi angka secara langsung melalui kamera, dilengkapi dengan opsi pemilihan model serta visualisasi hasil prediksi dalam bentuk diagram batang. Sementara itu, halaman deteksi *Grad-CAM* menyediakan visualisasi area fokus model dalam menentukan prediksi dengan menggunakan *heatmap Grad-CAM* sehingga pengguna dapat memahami bagian mana dari citra tangan yang paling berkontribusi terhadap keputusan model.

Sebelum *Dashboard* diimplementasikan, seluruh rancangan antarmuka divisualisasikan terlebih dahulu dalam bentuk *Wireframe*. Pembuatan *Wireframe* bertujuan untuk memberikan gambaran awal mengenai penyusunan elemen tampilan, alur interaksi pengguna, serta tata letak fitur yang akan dikembangkan. Melalui visualisasi ini, proses perancangan dapat dilakukan secara lebih terarah dan sistematis sehingga meminimalkan perubahan besar pada tahap implementasi.

2.6.1 Halaman Beranda

Wireframe halaman beranda pada *Dashboard* deteksi angka pada tangan menunjukkan susunan elemen yang membagi tampilan menjadi dua bagian utama. Pada bagian kiri terdapat panel navigasi yang menyediakan akses menuju halaman Deteksi *Realtime* dan Deteksi *Grad-CAM*. Panel ini berfungsi sebagai elemen navigasi utama bagi pengguna untuk berpindah halaman dengan mudah. Sementara itu, bagian kanan halaman menampilkan judul *Dashboard* beserta deskripsi singkat mengenai sistem, kemudian dilanjutkan dengan referensi bentuk tangan (0–9) yang berisi contoh gesture dari setiap angka yang akan dikenali oleh model. Selain informasi tersebut, juga ditampilkan tombol mulai prediksi yang mengarahkan pengguna menuju halaman deteksi *realtime* atau deteksi *Grad-CAM*. Tampilan *Wireframe* halaman ini dapat dilihat pada Gambar 10.

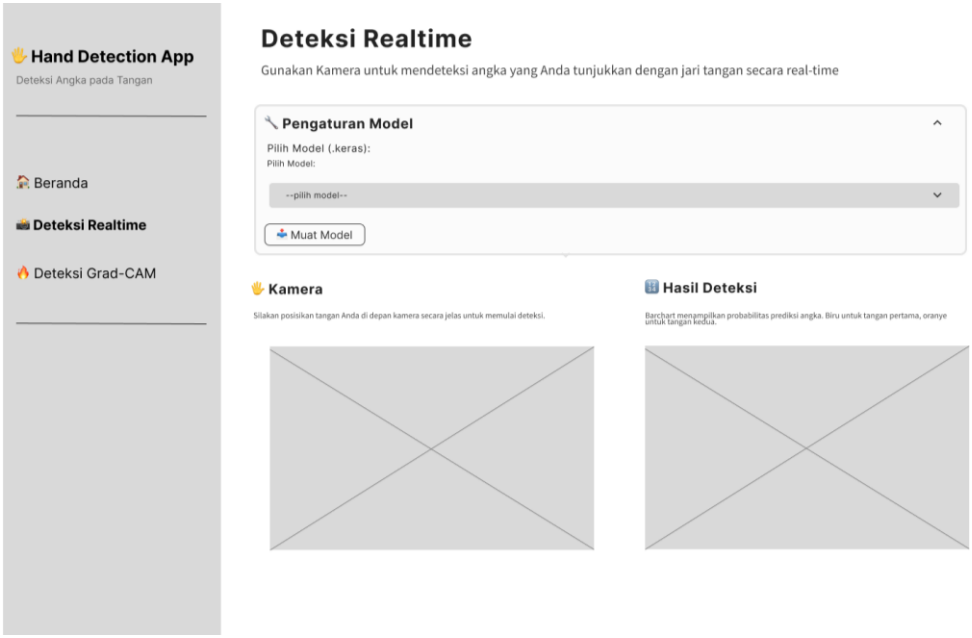


Gambar 10. *Wireframe* - halaman beranda

2.6.2 Halaman Deteksi *Realtime*

Wireframe halaman deteksi *realtime* dirancang sebagai pusat interaksi utama antara pengguna dan sistem dalam melakukan prediksi angka secara langsung. Bagian kiri halaman kembali menampilkan panel navigasi yang berfungsi untuk mengakses halaman beranda, halaman deteksi *realtime*, maupun halaman deteksi *Grad-CAM*. Pada bagian kanan halaman, komponen utama sistem ditampilkan secara teratur, dimulai dari bagian pengaturan model yang berupa dropdown untuk memilih file model (.keras) sebelum prediksi dijalankan. Di bawahnya terdapat tampilan kamera yang akan

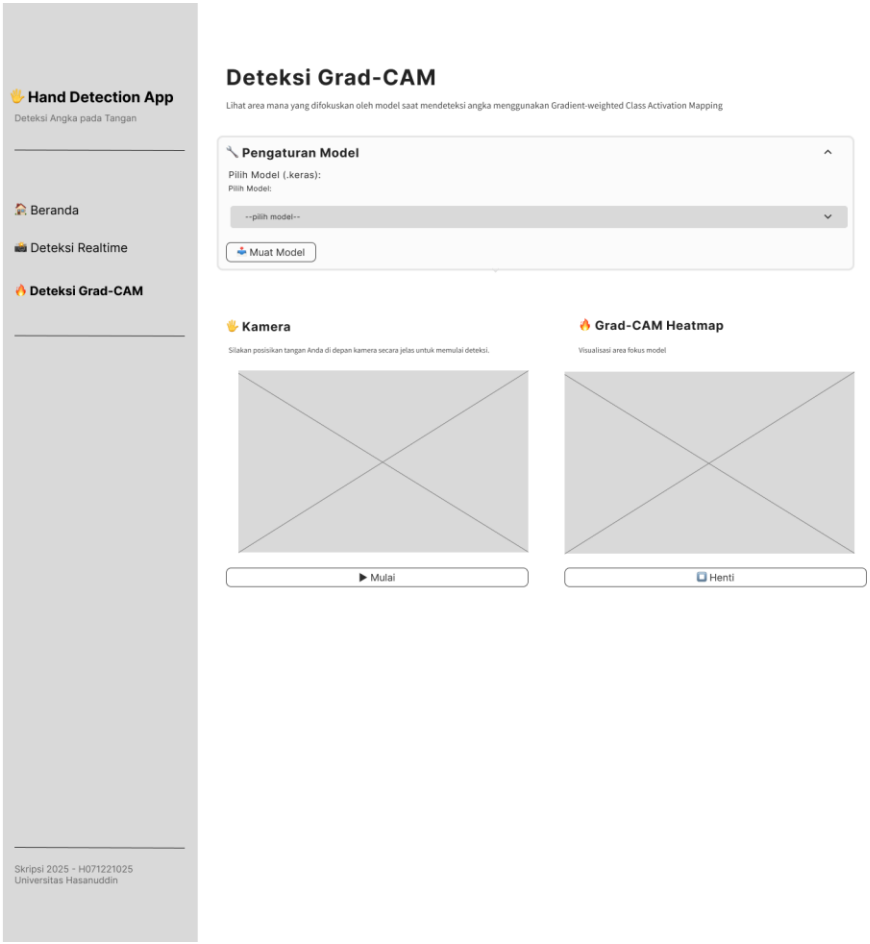
digunakan untuk menangkap gesture tangan secara *realtime*. Hasil prediksi dari model kemudian ditampilkan dalam bentuk diagram batang (barchart) yang menggambarkan probabilitas untuk setiap kelas angka yang terdeteksi. Susunan tampilan ini bertujuan agar proses deteksi dapat dilakukan dengan lebih intuitif dan hasilnya dapat dipahami dengan mudah oleh pengguna. *Wireframe* halaman deteksi *realtime* ditunjukkan pada Gambar 11.



Gambar 11. *Wireframe* - halaman deteksi *realtime*

2.6.3 Halaman Deteksi *Grad-CAM*

Wireframe halaman deteksi *Grad-CAM* memiliki struktur tampilan yang serupa dengan halaman deteksi *realtime*, namun dilengkapi dengan visualisasi *Grad-CAM* untuk menunjukkan area fokus model ketika melakukan prediksi. Bagian kiri halaman menampilkan panel navigasi yang tetap berfungsi untuk berpindah antarhalaman. Pada bagian kanan, pengguna dapat memilih model melalui dropdown pada bagian pengaturan model sebelum proses visualisasi dijalankan. Di bawah bagian tersebut terdapat tampilan kamera untuk menangkap citra tangan secara *realtime*, kemudian disampingnya ditampilkan *Grad-CAM* heatmap yang menunjukkan area mana pada citra tangan yang diperhatikan oleh model dalam menentukan prediksi. Visualisasi ini memberikan pemahaman tambahan mengenai bagaimana model membuat keputusan dan bagian mana dari citra yang paling berpengaruh dalam proses prediksi. Tampilan lengkap *Wireframe* halaman ini dapat dilihat pada Gambar 12.



Gambar 12. *Wireframe* - halaman deteksi *Grad-CAM*