BAB 2
LANDASAN TEORI

2.1 Penelitian Terdahulu

Berdasarkan penelitian terdahulu, upaya deteksi kardiomegali melalui citra rontgen dada telah dilakukan menggunakan pengembangan metode CNN. Penelitian oleh Alika Rahmarsyarah Rizalde (2025) menguji arsitektur CNN DenseNet169, InceptionResNetV2, dan MobileNetV2 dengan optimizer Adagrad, Nadam, dan SGD. Hasilnya menunjukkan bahwa kombinasi DenseNet169 + Nadam memberikan akurasi tertinggi yaitu 90%, diikuti MobileNetV2 + Nadam sebesar 87%, sehingga DenseNet169 dinilai sebagai konfigurasi paling optimal dalam mendeteksi pola citra kardiomegali.[17]. I Ketut Oning Pusparama dan I Putu Gede Hendra Suputra (2023) mengimplementasikan CNN untuk klasifikasi penyakit jantung dan memperoleh akurasi di atas 80%[18]. Estela Ribeiro dkk. (2023) menambahkan metode interpretabilitas dengan metode *explainability* seperti Grad-CAM, LIME, dan SHAP untuk membantu pemahaman hasil deteksi dengan akurasi 91,8%, presisi 74,0%, sensitivitas 87,0%, spesifisitas 92,9%, F1-score 79,8%, dan AUROC 90,0%, seluruh metode interpretabilitas kecuali SHAP mampu mengidentifikasi lokasi kardiomegali secara tepat, dan di antara ketiganya, Grad-CAM memiliki waktu komputasi yang lebih cepat dibandingkan LIME dan SHAP. [19].

Tabel 2.1. Penelitian Terdahulu

Deskripsi	**Bahtiar Waskito, dkk (2022)** [12]	**Alimi (2024)** [13]	**Fan et al. (2024)** [14]	**Penelitian Ini**
Algoritma CNN	v	v	-	v
Kardiomegali	v	-	v	v
Tingkat Akurasi >80%	v	v	v	v
Dataset >5000	v	-	v	v
Implementasi Website	-	v	-	v

Selain itu, A. M. Ayalew dkk. (2024) menggunakan *transfer learning* dengan ResNet-50, DenseNet-169, dan Inception, dilengkapi augmentasi data serta pemrosesan citra (BM3D dan CLAHE) dan Grad-CAM untuk visualisasi. Model ResNet-50 menunjukkan performa terbaik dengan akurasi 100% pada data latih dan validasi, 99,8% pada data uji, serta presisi, *recall*, dan F1-score masing-masing 100%, menegaskan potensinya untuk deteksi kardiomegali secara akurat [20]. K.-H. Lee (2024) mengembangkan model DNCD berbasis DenseNet121 untuk deteksi kardiomegali, dengan akurasi 95%, recall 91%, dan F1-score

94%. Model tersebut divalidasi menggunakan teknik interpretabilitas visual seperti *guided backpropagation* dan *saliency mapping*, yang menunjukkan bahwa area yang disorot model relevan dengan lokasi kardiomegali. Hasil penelitian ini menunjukkan bahwa model AI berkualitas tinggi dapat dikembangkan meskipun menggunakan jumlah data yang lebih sedikit dibandingkan studi sebelumnya, sehingga memperkuat potensi penerapan metode ini dalam bidang medis [21]. Selain itu, Sarpotdar (2022) menyajikan model U-Net kustom berbasis *deep learning* untuk mendeteksi kardiomegali pada citra X-ray dada. Penelitian ini menggunakan dataset *open source* "ChestX-ray8" dengan tahapan *preprocessing*, peningkatan citra, kompresi, dan klasifikasi sebelum proses pelatihan guna mengurangi waktu komputasi. Model U-Net yang dikembangkan mampu mencapai akurasi 94%, sensitivitas 96,2%, dan spesifisitas 92,5%, yang melampaui hasil model *pre-trained* sebelumnya dalam tugas deteksi kardiomegali [22]. Jain (2024) menunjukkan bahwa CNN efektif dalam memprediksi kardiomegali pada citra medis dengan metode klasifikasi biner, membedakan antara kasus kardiomegali dan non-kardiomegali. *Dataset* yang digunakan berjumlah 1114 sampel dengan distribusi seimbang, dan model mencapai akurasi keseluruhan 80%, dengan presisi, *recall*, dan F1-score yang relatif seimbang di kedua kelas [23].

2.2 Sistem Deteksi

Sistem deteksi adalah suatu mekanisme yang digunakan untuk memeriksa objek tertentu dengan metode atau teknik tertentu yang disusun dalam suatu sistem khusus. Sistem ini dirancang untuk menyelesaikan permasalahan tertentu, di mana cara penyelesaiannya tergantung pada metode yang diterapkan, sehingga dapat menghasilkan *output* yang diinginkan. Contoh penerapannya antara lain mendeteksi tingkat kematangan buah, menemukan kerusakan pada suatu barang, atau mengidentifikasi penyakit. Dalam konteks deteksi penyakit, sistem akan mengenali gejala atau tanda khusus yang berkaitan dengan penyakit tersebut, yang kemudian dijadikan parameter untuk menentukan hasil deteksinya [13].

Biasanya, sistem deteksi bekerja dengan menjalankan suatu algoritma, yaitu sekumpulan aturan yang dijalankan melalui proses perhitungan oleh sistem komputer. Algoritma tersebut bertujuan untuk menghimpun data atau informasi dari objek yang sedang diamati. Data tersebut dapat diperoleh dari berbagai perangkat, seperti *smartphone* atau laptop. Setelah dikumpulkan, informasi ini dibandingkan dengan data yang tersimpan dalam *database* menggunakan algoritma yang sama.

Hasil perbandingan tersebut kemudian menentukan *output* akhir dari sistem deteksi [13].

2.3 Penyakit Jantung Kardiomegali

Kardiomegali merupakan istilah dalam dunia medis yang digunakan untuk menggambarkan kondisi pembesaran ukuran jantung, yang dapat muncul sebagai akibat dari berbagai gangguan kesehatan. Kondisi ini tidak dikategorikan sebagai penyakit utama, melainkan sebagai indikasi adanya kelainan atau penyakit lain yang mendasarinya. Pembesaran jantung atau kardiomegali dapat dipicu oleh sejumlah faktor, di antaranya:

a. Penyakit Jantung Koroner: Penyempitan pada pembuluh darah koroner dapat menghambat aliran darah ke jantung, sehingga organ tersebut harus bekerja lebih berat.

b. Hipertensi (Tekanan Darah Tinggi): Kondisi tekanan darah yang meningkat secara kronis menyebabkan jantung memompa darah dengan tenaga lebih besar, yang berpotensi memicu pembesaran jantung.

c. Penyakit Katup Jantung: Gangguan atau kerusakan pada katup jantung dapat mengurangi efisiensi proses pemompaan darah.

d. Kardiomiopati: Sekelompok kelainan pada otot jantung yang dapat berkontribusi terhadap terjadinya kardiomegali.

e. Kondisi Lain: Termasuk anemia berat, infeksi tertentu, serta gangguan pada kelenjar tiroid [24].

Citra rontgen dada pada pasien dengan kardiomegali dapat dikenali melalui adanya peningkatan ukuran jantung yang tampak lebih besar dibandingkan dengan kondisi normal, terutama jika dilihat dari rasio jantung terhadap rongga dada. Pembesaran tersebut umumnya ditunjukkan oleh melewati batas anatomis normal jantung pada foto rontgen posteroanterior, sehingga menjadi indikator awal adanya kelainan kardiovaskular. Selain itu, pemeriksaan klinis tambahan seperti anamnesis, pemeriksaan fisik, elektrokardiogram (EKG), dan ekokardiografi juga berperan penting dalam menunjang dan mengonfirmasi proses diagnosis, sebagaimana ditampilkan pada Gambar 2.1.

Gambar 2.1. Foto rontgen pasien dengan kardiomegali.
Sumber: [25]

2.4 Citra Digital

Citra dapat direpresentasikan sebagai suatu fungsi intensitas dua dimensi $f(x,y)$, dengan x dan y sebagai koordinat spasial, sementara nilai $f(x,y)$ menunjukkan tingkat kecerahan *(brightness)* pada titik tersebut. Citra digital diperoleh melalui proses digitalisasi terhadap fungsi citra $f(x,y)$, baik pada aspek koordinat spasial maupun pada level kecerahan *(brightness level)*. Nilai fungsi f pada posisi (x,y) menggambarkan derajat keabuan *(grayness)* atau intensitas cahaya citra pada lokasi tersebut.

Secara umum, citra digital dapat didefinisikan sebagai representasi citra dua dimensi yang tersusun atas sekumpulan nilai digital yang dikenal sebagai elemen gambar atau *pixel*. Setiap *pixel* merupakan unit terkecil dalam citra yang menyimpan informasi warna maupun tingkat kecerahan pada suatu lokasi tertentu. Pada umumnya, citra digital berbentuk persegi panjang atau bujur sangkar, namun pada sistem pencitraan tertentu dimungkinkan dihasilkan citra dengan struktur segi enam. Dimensi citra, yang mencakup lebar dan tinggi, dinyatakan dalam jumlah *pixel* sehingga bernilai bilangan bulat. Setiap *pixel* memiliki koordinat posisi yang unik di dalam citra, yang umumnya direpresentasikan dalam bilangan bulat positif dan dapat dimulai dari 0 atau 1 sesuai dengan sistem koordinat yang digunakan. Selain itu, setiap *pixel* memiliki nilai digital yang merepresentasikan informasi citra pada titik tersebut, khususnya warna atau tingkat kecerahan.

Representasi warna pada citra digital sangat ditentukan oleh format citra yang digunakan. Dalam berbagai aplikasi visual, nilai digital pada setiap *pixel* digunakan untuk merepresentasikan warna citra. Beberapa format citra digital yang umum dijumpai antara lain citra biner (*monokrom*), citra skala keabuan (*grayscale*), citra berwarna (*true color*), serta citra warna berindeks (*indexed color*).

Pengolahan citra merupakan suatu tahapan manipulasi citra yang memanfaatkan citra sebagai masukan (*input*) untuk menghasilkan keluaran (*output*) berupa citra baru atau sekumpulan parameter yang berkaitan dengan citra tersebut. Dalam konteks digital, pengolahan citra merujuk pada proses pemrosesan citra dua dimensi dengan bantuan komputer. Citra digital sendiri dapat disajikan dalam berbagai bentuk visual, seperti citra biner, citra skala keabuan (*grayscale*), dan citra berwarna [26].

1. Citra Biner

   Citra biner merupakan jenis citra digital yang hanya memiliki dua kemungkinan nilai warna, yaitu hitam dan putih. Kedua warna tersebut dinyatakan dalam bentuk nilai biner, yakni 0 dan 1. Pada citra biner, penafsiran terhadap nilai biner dapat dilakukan melalui dua model pendekatan, yaitu sebagai berikut:

   Model cahaya: Nilai 1 merepresentasikan warna putih yang menunjukkan keberadaan cahaya, sedangkan nilai 0 menggambarkan warna hitam yang menandakan tidak adanya cahaya.

   Model tinta/cat: Nilai 1 diartikan sebagai warna hitam yang menunjukkan keberadaan tinta atau cat, sedangkan nilai 0 menunjukkan warna putih yang menandakan ketiadaan tinta.

2. Citra *Grayscale*

   Citra *grayscale* merupakan bentuk citra digital yang menyatakan intensitas piksel dalam variasi tingkat keabuan. Banyaknya tingkat keabuan ditentukan oleh kedalaman bit yang digunakan, yaitu sesuai dengan hasil perpangkatan dua dari jumlah bit tersebut. Sebagai ilustrasi, citra dengan kedalaman 4-bit memiliki 16 tingkat keabuan dengan nilai berkisar antara 0 hingga 15, di mana nilai 0 merepresentasikan warna hitam, nilai maksimum 15 menunjukkan warna putih, dan nilai di antaranya menggambarkan gradasi abu-abu dengan tingkat kecerahan yang berbeda.

3. Citra Warna

   Citra warna merupakan bentuk representasi citra digital yang dibangun dari kombinasi tiga warna dasar, yaitu merah, hijau, dan biru (*RGB*). Setiap *pixel* pada citra warna menyimpan informasi intensitas dari ketiga komponen warna tersebut, dengan rentang nilai masing-masing antara 0 hingga 255 (8-bit). Pada kanal merah (*Red*), nilai minimum menunjukkan warna putih dan nilai maksimum menunjukkan warna merah. Pada kanal hijau (*Green*), nilai minimum menunjukkan warna putih dan nilai maksimum menunjukkan warna hijau. Sementara itu, pada kanal biru (*Blue*), nilai minimum menunjukkan warna putih dan nilai maksimum menunjukkan warna biru.

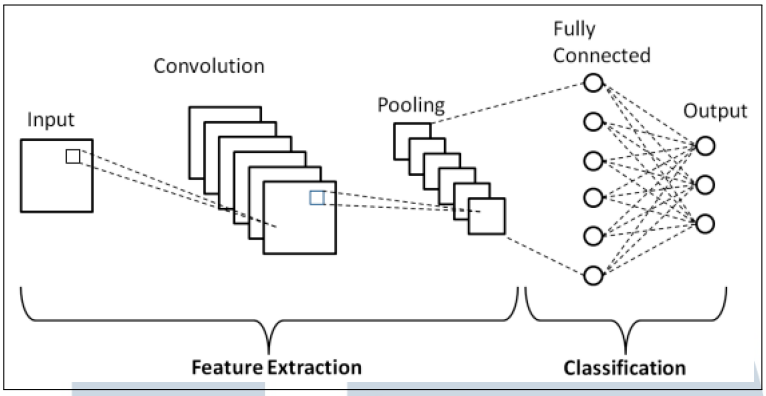
   Sebagai ilustrasi, warna ungu dihasilkan dari kombinasi kanal merah dan biru, sehingga memiliki nilai RGB sebesar (255, 0, 255). Apabila ketiga kanal warna pada suatu *pixel* bernilai minimum, maka warna yang terbentuk adalah hitam. Transformasi warna dapat dilakukan untuk mengubah model representasi warna, seperti dari RGB ke YUV atau ke *grayscale*. Selain itu, prinsip ortonormal dapat diterapkan pada dekomposisi warna, di mana tiga kanal pada setiap *pixel* dapat menghasilkan representasi sinyal yang jarang dengan hanya satu koefisien tidak bernilai nol dari ketiga kanal tersebut [26].

2.5 CNN (Convolutional Neural Network)

CNN *(Convolutional Neural Network)* merupakan salah satu arsitektur jaringan saraf yang memanfaatkan lapisan konvolusi untuk mengolah data visual secara efektif. Pada umumnya, CNN memiliki dua tipe *hidden layer*, yaitu *convolution layer* dan *pooling layer*, yang disusun secara berurutan dan saling bergantian. Berbeda dengan jaringan saraf konvensional, CNN menerapkan konsep *weight sharing* sehingga jumlah parameter yang digunakan dapat ditekan secara signifikan. Pendekatan ini berkontribusi dalam mengurangi kompleksitas model sekaligus meningkatkan efisiensi waktu pelatihan. Dalam pemrosesan citra sebagai *input*, CNN mampu menerima gambar secara langsung tanpa memerlukan tahapan ekstraksi fitur secara manual, karena jaringan ini dapat mempelajari pola visual secara otomatis melalui mekanisme *local receptive field*, *convolution*, dan *pooling*. Dengan karakteristik tersebut, CNN menunjukkan ketahanan yang baik terhadap berbagai transformasi citra, seperti pergeseran posisi, rotasi, maupun perubahan skala [27].

Dalam arsitektur CNN yang digambarkan pada Gambar 2.2, proses pengolahan citra dimulai dari lapisan input, di mana citra beresolusi tertentu (1024 x 1024 piksel atau 2048 x 2048 piksel, tergantung alat radiografi) diterima sebagai data mentah yang akan dianalisis. Selanjutnya, citra dilewatkan ke serangkaian lapisan konvolusi *(convolution layers)* yang bertanggung jawab mengekstraksi fitur lokal menggunakan *filter* atau kernel yang bergerak di seluruh piksel gambar untuk mendeteksi pola seperti tepi, tekstur, dan bentuk khusus [28]. Hasil keluaran dari lapisan konvolusi kemudian diteruskan ke lapisan *pooling*, yang berfungsi melakukan reduksi dimensi spasial sambil mempertahankan fitur penting, sehingga ukuran data berkurang dan perhitungan menjadi lebih efisien tanpa kehilangan informasi representatif [29]. Tahapan berikutnya adalah lapisan FC *(fully connected)*, di mana neuron-neuron saling terhubung secara penuh untuk menggabungkan fitur-fitur yang telah diekstraksi menjadi representasi yang lebih global dan kemudian menghasilkan output klasifikasi akhir berdasarkan pola yang dipelajari selama pelatihan [30]. Secara keseluruhan, CNN mampu mempelajari pola visual secara hierarkis melalui kombinasi ekstraksi fitur dan klasifikasi, sehingga sangat cocok untuk tugas klasifikasi citra medis seperti deteksi kardiomegali dari citra rontgen [31].

Dalam penelitian ini, pemilihan teknik augmentasi disesuaikan untuk mensimulasikan kondisi pengamatan yang berbeda-beda dan meningkatkan variasi visual dari *dataset*. Teknik augmentasi yang digunakan meliputi RandomHorizonalFlip untuk menciptakan simetri horizontal acak, RandomRotation untuk menambahkan variasi orientasi, yaitu perubahan sudut atau posisi kemiringan citra, dan ColorJitter untuk memvariasikan kecerahan, kontras, atau saturasi cahaya [32]. Sebelum proses augmentasi dilakukan, citra melewati tahap pra-pemrosesan terlebih dahulu seperti Resize untuk menyamakan dimensi input ke ukuran yang dibutuhkan oleh CNN, diikuti oleh Normalize untuk menyesuaikan nilai piksel ke rentang standar. Oleh karena itu, augmentasi dan pra-pemrosesan dilakukan untuk membuat model lebih baik dalam membedakan (mengklasifikasikan) citra dan memastikan fitur yang dipelajari tetap akurat meskipun citra mengalami perubahan [33].



Gambar 2.2. Arsitektur CNN
Sumber: [34]

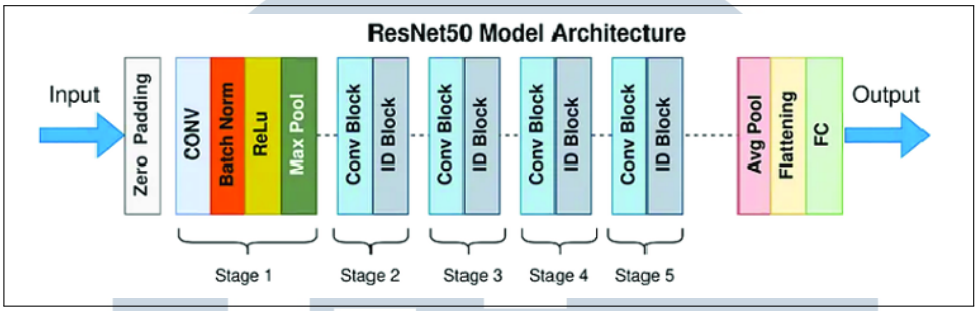
2.6 ResNet-50

Arsitektur ResNet-50 merupakan salah satu jaringan *convolutional* berlapis dalam dengan total 50 lapisan, yang dirancang menggunakan konsep *residual connection* untuk mengatasi penurunan performa ketika kedalaman jaringan meningkat. Melalui mekanisme tersebut, ResNet-50 dapat belajar mengekstraksi pola dan fitur visual yang kompleks dari citra tanpa mengalami kendala *vanishing gradient*. Dalam penelitian ini, ResNet-50 berperan sebagai ekstraksi fitur yang mengekstraksi informasi spasial mendalam dari citra rontgen dada. Hasil ekstraksi tersebut kemudian diteruskan ke lapisan klasifikasi guna membedakan antara citra yang tergolong normal dan citra yang menunjukkan indikasi kardiomegali. [35].

2.6.1 Arsitektur ResNet-50

Model ResNet-50 merupakan salah satu jaringan *convolutional* yang berlapis dalam, terdiri atas 50 lapisan, dan menggunakan konsep *residual learning* untuk mempermudah proses pelatihan pada jaringan yang memiliki kedalaman tinggi. Gambar 2.3 memperlihatkan alur kerja lengkap arsitektur ResNet-50, mulai dari tahap masukan *(input)* hingga menghasilkan keluaran *(output)*. Gambar 2.3 menampilkan alur pemrosesan data pada arsitektur ResNet-50, yang diawali dengan tahap pra pemrosesan berupa *zero padding* untuk mempertahankan dimensi spasial citra, diikuti oleh lapisan konvolusi awal dengan filter berukuran besar guna mengekstraksi fitur dasar. Hasil konvolusi kemudian dinormalisasi menggunakan *batch normalization* untuk menjaga stabilitas distribusi data selama

proses pelatihan, dilanjutkan dengan penerapan fungsi aktivasi ReLU sebagai pemetaan non linear, serta *max pooling* untuk mereduksi dimensi spasial tanpa menghilangkan informasi penting.



Gambar 2.3. Arsitektur ResNet-50 *general* dari lapisan *input* sampai klasifikasi *output*
Sumber: [36]

Selanjutnya, data diproses melalui lima tahapan utama (*stage* dua sampai lima) yang terdiri atas kombinasi *convolutional block* dan *identity block*. *Convolutional block* berfungsi menyesuaikan dimensi input agar sesuai dengan struktur jaringan berikutnya, sedangkan *identity block* mempertahankan ukuran fitur dan memungkinkan adanya *shortcut connection* guna memperlancar propagasi gradien antar lapisan. Mekanisme propagasi gradien dalam blok residual didukung oleh *shortcut connection* yang menciptakan jalur ganda. Selama proses *backpropagation*, yaitu tahap pembaruan bobot berdasarkan kesalahan prediksi, gradien kesalahan tidak hanya dialirkan melalui jalur konvolusi utama, tetapi juga melalui *shortcut connection*. Jalur pintas ini menyediakan lintasan langsung dengan kontribusi identitas, sehingga gradien dapat mengalir tanpa mengalami pelemahan yang signifikan. Mekanisme ini berperan penting dalam mencegah terjadinya *vanishing gradient* pada jaringan dengan kedalaman yang besar. Dengan cara ini, sinyal gradien dijamin dapat mengalir secara efisien dan kuat hingga ke lapisan awal ResNet, sehingga memungkinkan pelatihan model dengan kedalaman ekstrem tanpa mengalami degradasi performa.

Pada tahap akhir, dilakukan *average pooling* untuk memperoleh representasi global dari fitur *map*, diikuti oleh *flattening* yang mengubah data menjadi bentuk vektor satu dimensi. Proses klasifikasi kemudian dilakukan melalui *FCL (fully connected layer)* yang pada umumnya menggunakan fungsi aktivasi *Softmax* guna menghitung probabilitas masing-masing kelas pada *output*. Struktur *residual block* dengan *shortcut connection* pada arsitektur ini memungkinkan pelatihan jaringan yang lebih dalam tanpa mengalami degradasi performa, karena gradien dapat

mengalir secara lebih efisien ke lapisan awal selama proses pelatihan.

Pengaturan parameter pelatihan pada arsitektur ResNet-50 dilakukan dengan menyeragamkan ukuran citra (*IMG_SIZE*) sehingga seluruh citra rontgen memiliki dimensi masukan yang konsisten. Proses pelatihan menggunakan nilai *batch size* yang disesuaikan untuk menjaga keseimbangan antara efisiensi komputasi dan kestabilan pembaruan bobot jaringan. Nilai laju pembelajaran (*learning rate*) ditetapkan relatif kecil guna memastikan proses konvergensi berlangsung secara stabil. *Optimizer* yang digunakan adalah *Adam*, karena kemampuannya dalam melakukan penyesuaian laju pembelajaran secara adaptif selama pelatihan. Sementara itu, fungsi kerugian yang diterapkan berupa *categorical cross entropy* yang berfungsi untuk mengukur selisih antara label aktual dan hasil prediksi model. Jumlah *epoch* ditentukan secara bertahap hingga model mencapai kinerja optimal tanpa mengalami kondisi *overfitting* [36].

2.6.2 Convolution Layer

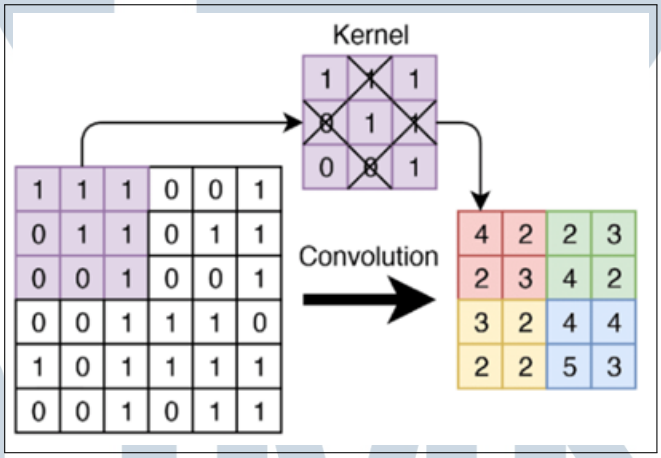
Dalam arsitektur ResNet-50, proses ekstraksi fitur dari citra masukan dilakukan melalui lapisan konvolusi *(Convolutional Layer)* yang berfungsi mempertahankan keterkaitan spasial antar piksel sekaligus mempelajari pola visual menggunakan filter yang dapat dioptimalkan selama pelatihan. Model ini juga memperkenalkan konsep *cardinality*, yaitu jumlah jalur paralel yang memproses informasi secara bersamaan di dalam satu blok *residual*. Pendekatan tersebut memungkinkan jaringan memperoleh representasi fitur yang lebih kaya tanpa harus meningkatkan jumlah parameter secara signifikan [36]. Rumus penjumlahan residual di ResNet-50 yang dapat dilihat pada Persamaan 2.1 menggambarkan mekanisme penjumlahan residual yang digunakan dalam jaringan saraf.

$$y = x + f(x) \tag{2.1}$$

- x adalah *input* awal yang masuk ke blok, sekaligus menjadi jalur pintas (*shortcut connection*) yang langsung ditambahkan ke *output* akhir.
- $f(x)$ adalah fungsi transformasi atau lapisan-lapisan yang dipelajari (biasanya terdiri dari lapisan konvolusi, *normalisasi batch*, dan aktivasi ReLU). Fungsi ini bertugas mempelajari *residual mapping* atau pemetaan residual, yaitu penyesuaian yang diperlukan di atas input awal x.

- y adalah *output* akhir dari blok residual.

Gambar 2.4 menampilkan konsep fundamental dari proses konvolusi pada sebuah *convolution layer*. Matriks yang terletak di sisi kiri menggambarkan citra sebagai data masukan, sedangkan matriks berukuran lebih kecil yang berada di atasnya berperan sebagai kernel atau filter dalam operasi konvolusi. Kernel ini digeser secara sistematis ke seluruh bagian citra, dan pada setiap posisi dilakukan perkalian elemen demi elemen antara nilai kernel dan area citra yang bersesuaian, kemudian hasilnya dijumlahkan. Nilai akhir dari proses tersebut menghasilkan elemen baru yang menyusun matriks keluaran konvolusi di sisi kanan. Melalui mekanisme ini, jaringan saraf tiruan dapat mengekstraksi serta mengenali fitur-fitur penting seperti tepi, tekstur, dan pola visual pada citra, yang selanjutnya dimanfaatkan dalam proses klasifikasi maupun pengenalan objek [37].

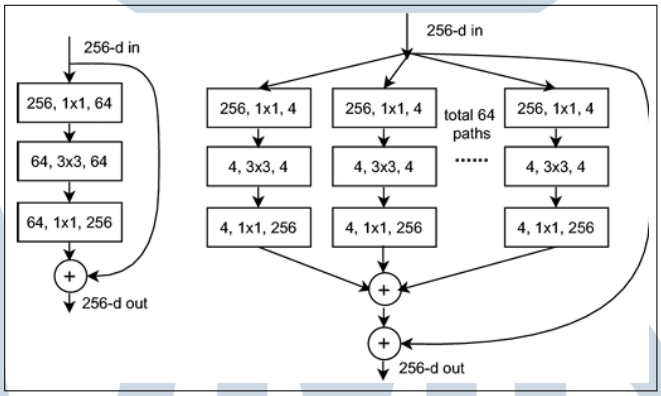


Gambar 2.4. Cara kerja *convolutional layer*
Sumber: [37]

2.6.3 Bottleneck Block

Komponen inti dari arsitektur ResNet-50 adalah *Bottleneck Block*, yang dirancang dengan beberapa jalur konvolusi paralel berskala kecil untuk meningkatkan efisiensi dibandingkan metode konvolusi konvensional. Setiap jalur menjalankan operasi konvolusi dengan konfigurasi serupa, kemudian hasilnya digabungkan melalui proses penjumlahan guna menghasilkan *output* akhir. Struktur ini mempertahankan konsep *deep residual learning* yang menjadi ciri khas ResNet-50, namun dengan jumlah parameter yang lebih sedikit serta kinerja komputasi yang lebih optimal [36].

Gambar 2.5 menampilkan dua jenis struktur blok residual pada arsitektur ResNet-50, yaitu *bottleneck* konvensional (kiri) dan versi dengan tingkat *cardinality* lebih tinggi (kanan). Pada desain *bottleneck* standar, data masukan berdimensi 256 melewati tiga lapisan konvolusi secara berurutan: konvolusi 1×1 untuk mengurangi dimensi, dilanjutkan dengan konvolusi 3×3, lalu diakhiri dengan konvolusi 1×1 untuk mengembalikan dimensi semula menjadi 256. Hasil dari proses ini kemudian dijumlahkan dengan *input* awal melalui *shortcut connection*. Sementara itu, pada struktur di sisi kanan diterapkan konsep *cardinality*, di mana masukan diproses secara paralel melalui 64 jalur konvolusi yang identik. Setiap jalur memiliki tiga lapisan konvolusi yang sama, tetapi dengan ukuran dimensi yang lebih kecil (contohnya dari 256 menjadi 4), sehingga komputasi menjadi lebih efisien. Seluruh keluaran dari jalur-jalur tersebut kemudian digabungkan dan ditambahkan kembali ke *input* awal untuk menghasilkan *output* akhir berdimensi 256. Pendekatan ini memperkaya kemampuan jaringan dalam menangkap variasi fitur tanpa menambah jumlah parameter secara signifikan, sehingga memperkuat representasi fitur yang lebih kompleks dan informatif [38].



Gambar 2.5. Struktur bottleneck block residual block pada model ResNet-50
Sumber: [38]

2.6.4 Aggregation Layer

Dalam arsitektur ResNet-50, *Aggregation Layer* memiliki peran penting dalam menyatukan keluaran dari beberapa jalur transformasi paralel yang terdapat pada blok *residual*. Setiap jalur melakukan operasi konvolusi terhadap input yang sama, kemudian hasilnya digabungkan melalui proses penjumlahan untuk menghasilkan output akhir. Mekanisme ini memungkinkan jaringan untuk mempelajari beragam representasi fitur secara bersamaan tanpa menambah beban

komputasi yang besar. Dengan demikian, struktur paralel ini membantu proses ekstraksi informasi menjadi lebih efisien dan stabil dalam pembelajaran fitur citra medis [36].

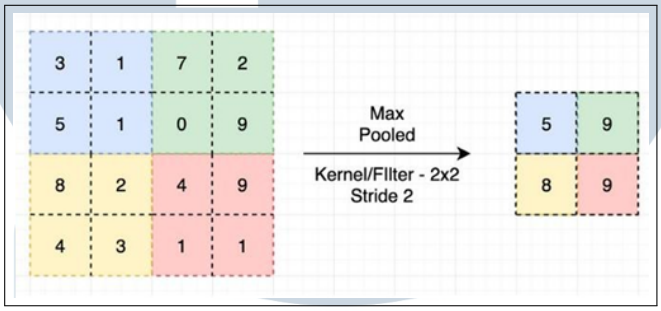
2.6.5 Pooling Layer

Pooling Layer berperan dalam mereduksi dimensi peta aktivasi tanpa menghilangkan informasi penting yang terkandung dalam citra. Tahapan ini dilakukan dengan membagi citra masukan ke dalam beberapa wilayah kecil yang saling terpisah, kemudian setiap wilayah tersebut disederhanakan nilainya menggunakan operasi nonlinier, seperti *max pooling* yang mengambil nilai terbesar atau *average pooling* yang menghitung nilai rata-rata. Keberadaan lapisan *pooling* memungkinkan jaringan untuk melakukan generalisasi pola secara lebih efektif, mempercepat proses konvergensi selama pelatihan, serta meningkatkan ketahanan terhadap pergeseran maupun distorsi posisi objek pada citra. Umumnya, lapisan ini ditempatkan di antara lapisan konvolusi guna menjaga efisiensi komputasi dan stabilitas proses pembelajaran [39]. Rumus 2.2 menggambarkan mekanisme perhitungan yang digunakan dalam *Pooling Layer* pada jaringan saraf.

$$p_{j,m} = \max(h_{j,(m-1)N+r}) \tag{2.2}$$

- $p_{j,m}$ adalah nilai *output* yang telah direduksi (*pooled*) pada posisi ke-m di peta fitur ke-j.
- $\max(\dots)$ adalah operator non-linear yang mengambil nilai tertinggi (maksimum) dari serangkaian *input* yang berada dalam jendela *pooling*.
- $h_{j,(m-1)N+r}$ merepresentasikan nilai pada peta aktivasi input h.
- j adalah indeks peta fitur *(channel)* yang sedang diproses.
- m adalah indeks posisi *output* yang dihasilkan.
- N adalah ukuran langkah (*stride*) yang digunakan untuk menggeser jendela *pooling*.
- r adalah indeks yang berjalan di sepanjang dimensi jendela *pooling*.

Gambar 2.6 menggambarkan mekanisme *max pooling* menggunakan filter berukuran 2×2 dengan nilai *stride* sebesar 2. Pada proses ini, citra masukan dipartisi ke dalam beberapa blok kecil berukuran 2×2 yang tidak saling tumpang tindih. Setiap blok kemudian direpresentasikan oleh satu nilai, yaitu nilai maksimum dari elemen-elemen yang terdapat di dalamnya. Sebagai ilustrasi, pada blok kiri atas yang memiliki elemen {3, 1, 5, 1}, nilai tertinggi yaitu 5 dipilih sebagai hasil. Prosedur ini diterapkan secara menyeluruh pada citra, sehingga menghasilkan citra keluaran dengan dimensi yang lebih kecil, namun tetap mempertahankan informasi dominan dari citra semula. Pendekatan ini terbukti efektif dalam menurunkan kompleksitas komputasi serta meningkatkan ketahanan model terhadap pergeseran posisi objek pada citra [40].



Gambar 2.6. Ilustrasi proses operasi *max pooling* pada lapisan *pooling layer*
Sumber: [40]

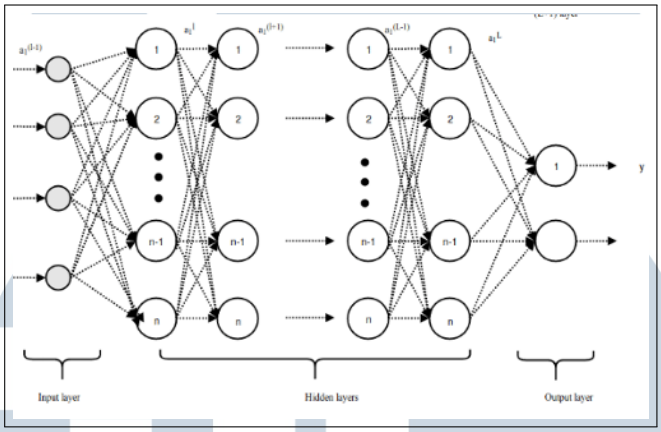
2.6.6 Fully Connected Layer

Lapisan *FCL (Fully Connected Layer)* dalam arsitektur ResNet-50 berperan sebagai komponen akhir yang melakukan proses klasifikasi berdasarkan fitur-fitur yang telah diperoleh dari lapisan residual sebelumnya. Pada tahap ini, setiap neuron memiliki koneksi penuh terhadap seluruh neuron pada lapisan sebelumnya, sehingga jaringan mampu mengintegrasikan berbagai representasi fitur yang telah dipelajari. Umumnya, lapisan ini menggunakan fungsi aktivasi *Softmax* untuk menghasilkan nilai probabilitas bagi setiap kelas yang diprediksi [41]. Rumus 2.3 berikut menggambarkan mekanisme perhitungan pada *Fully Connected Layer*.

$$a = \sigma(WX + b) \tag{2.3}$$

- *a* adalah aktivasi akhir atau *output* dari *neuron* FCL, yang digunakan untuk prediksi kelas.
- σ adalah fungsi aktivasi non-linear (seperti ReLU atau Softmax) yang diterapkan pada hasil *pre-activation*.
- *W* adalah matriks bobot (*weight matrix*). Matriks ini dipelajari selama pelatihan dan menentukan kekuatan pengaruh setiap fitur *input* terhadap *neuron* saat ini.
- *X* adalah vektor *input* yang berisi semua fitur yang diekstraksi dari lapisan-lapisan sebelumnya.
- *b* adalah vektor *bias* yang ditambahkan untuk menggeser (*shift*) nilai aktivasi dan memberikan fleksibilitas tambahan pada model.

Gambar 2.7 menggambarkan arsitektur dasar dari *FCL (Fully Connected Layer)* yang terdiri atas tiga komponen utama, yaitu lapisan *input*, lapisan tersembunyi *(hidden layers)*, dan lapisan *output*. Pada struktur ini, setiap neuron pada suatu lapisan terhubung sepenuhnya dengan seluruh neuron pada lapisan selanjutnya, yang direpresentasikan melalui garis penghubung antar neuron. Informasi dari lapisan *input* dialirkan ke lapisan tersembunyi, di mana setiap koneksi memiliki bobot *(weight)* dan bias, kemudian diproses menggunakan fungsi aktivasi seperti ReLU atau sigmoid. Alur pemrosesan ini diteruskan hingga mencapai lapisan *output*, yang umumnya menerapkan fungsi aktivasi *Softmax* untuk mengonversi hasil keluaran menjadi nilai probabilitas kelas. Struktur jaringan tersebut memungkinkan model untuk mempelajari dan merepresentasikan hubungan non linear yang kompleks antar fitur hasil ekstraksi dari lapisan sebelumnya [40].



Gambar 2.7. Arsitektur fully connected layer pada model ResNet-50
Sumber: [40]

2.6.7 Activation Function

Fungsi aktivasi memiliki peran krusial dalam arsitektur ResNet-50, karena berfungsi untuk menangani sifat non linearitas pada data sehingga jaringan mampu memecahkan permasalahan kompleks dan non trivial. Setiap fungsi aktivasi bekerja dengan menerima suatu nilai input dan menerapkan operasi matematis tertentu terhadapnya. Dalam struktur ResNet-50, fungsi aktivasi biasanya ditempatkan setelah proses konvolusi atau *pooling*, atau digunakan pada tahap akhir perhitungan fitur *map* untuk menghasilkan pola fitur yang lebih representatif. Jenis fungsi aktivasi yang diterapkan dalam ResNet-50 adalah *ReLU (Rectified Linear Unit)* [27]. Rumus 2.4 berikut menggambarkan fungsi dasar dari ReLU.

$$f(x) = \max(0, x) \tag{2.4}$$

- $f(x)$ adalah *output* atau nilai aktivasi *neuron* setelah fungsi diterapkan.
- x adalah *input* yang masuk ke fungsi aktivasi, yang biasanya merupakan hasil dari operasi penjumlahan berbobot dan *bias* ($WX + b$).
- $\max(0, x)$ adalah operator matematis yang menentukan nilai $f(x)$:
  - Jika $x > 0$, maka $f(x) = x$.
  - Jika $x \leq 0$, maka $f(x) = 0$.

2.7 Euclidean Distance

Euclidean Distance merupakan salah satu metrik jarak yang paling banyak digunakan dalam bidang pengenalan pola dan pengolahan citra. Pendekatan ini didasarkan pada pemodelan citra sebagai sebuah titik di dalam ruang Euclidean berdimensi tinggi (*high dimensional Euclidean space*), di mana setiap *pixel* direpresentasikan sebagai sebuah koordinat. Berdasarkan representasi tersebut, tingkat kemiripan antara dua citra dapat ditentukan dengan mengukur jarak antar titik pada ruang multidimensi. Semakin serupa nilai *pixel* pada kedua citra, maka semakin kecil nilai jarak Euclidean yang dihasilkan, bahkan dapat mendekati nol.

Dalam konteks pengolahan citra, suatu gambar dapat dinyatakan sebagai sebuah vektor dengan dimensi sebanyak n. Sebagai ilustrasi, citra dengan ukuran 10×10 piksel dapat dipetakan sebagai sebuah titik pada ruang berdimensi 100. Prinsip yang sama juga diterapkan pada citra berwarna (RGB), di mana citra direpresentasikan dalam bentuk tiga matriks dua dimensi yang masing-masing memuat informasi intensitas pada kanal merah, hijau, dan biru. Setelah citra tersebut dikonversi ke dalam format keabuan (*grayscale*), representasinya menjadi satu matriks dua dimensi dengan nilai piksel berada pada kisaran 0 sampai 255, di mana nilai yang lebih kecil merepresentasikan area citra yang lebih gelap [42]. Euclidean Distance digunakan untuk mengukur panjang garis lurus antara dua titik p dan q dalam ruang Euclidean. Jarak ini dihitung berdasarkan perbedaan nilai koordinat pada setiap dimensi dalam rumus 2.5 sebagai berikut:

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{2.5}$$

- p dan q adalah dua titik dalam ruang Euclidean berdimensi-n.
- p_i dan q_i adalah koordinat dari masing-masing titik pada dimensi ke-i.
- n adalah jumlah dimensi atau panjang vektor representasi citra.

2.8 Confusion Matrix

Konsep Confusion Matrix pertama kali diperkenalkan oleh Karl Pearson pada tahun 1904 dalam bentuk *contingency table*. Istilah ini kemudian berkembang menjadi *classification matrix* sebelum akhirnya dikenal sebagai *confusion matrix*

dalam disiplin ilmu data. Penggunaan istilah ”confusion” merujuk pada potensi kebingungan dalam menentukan metrik mana yang harus diprioritaskan saat meningkatkan kinerja model, mengingat berbagai metrik evaluasi dapat diperoleh dari matriks ini [43].

Confusion Matrix adalah sebuah matriks persegi berukuran $N \times N$ yang digunakan untuk merepresentasikan jumlah kelas keluaran (*output classes*) pada suatu model klasifikasi. Setiap baris pada matriks menunjukkan jumlah data berdasarkan hasil prediksi model, sedangkan setiap kolom merepresentasikan jumlah data berdasarkan kelas sebenarnya. Penyajian ini memberikan informasi yang detail mengenai jumlah prediksi yang tepat maupun keliru pada masing-masing kelas, baik untuk permasalahan klasifikasi biner maupun *multiclass*. Selain itu, confusion matrix dimanfaatkan untuk menilai kinerja model, menganalisis pola kesalahan yang terjadi, serta menentukan strategi perbaikan model selanjutnya [43]. Karena pembentukan confusion matrix memerlukan data dengan label atau nilai target yang telah diketahui, metode ini hanya diterapkan pada pendekatan *supervised learning*. Tabel berikut memperlihatkan contoh confusion matrix pada kasus klasifikasi biner:

Tabel 2.2. Confusion Matrix

Predicted Values	**Actual Positive**	**Actual Negative**
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

Confusion matrix yang dapat dilihat pada Tabel 2.2 memiliki empat jenis kemungkinan hasil prediksi, yaitu True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN). Nilai TP menggambarkan kondisi ketika data yang diprediksi sebagai positif sesuai dengan kondisi sebenarnya yang juga positif, sedangkan TN menunjukkan data yang diprediksi negatif dan benar-benar berada pada kelas negatif. FP terjadi apabila model memberikan prediksi positif pada data yang sebenarnya negatif, yang dikenal sebagai kesalahan Tipe I. Sementara itu, FN muncul ketika model memprediksi negatif pada data yang sebenarnya positif, yang disebut sebagai kesalahan Tipe II. Pada istilah TP, TN, FP, dan FN, bagian pertama *(True* atau *False)* menunjukkan tingkat kebenaran prediksi, sedangkan bagian kedua *(Positive* atau *Negative)* merepresentasikan kelas hasil prediksi yang dihasilkan oleh model.

2.8.1 Accuracy

Akurasi merupakan rasio prediksi yang benar dibandingkan seluruh prediksi. Metrik ini umum digunakan untuk mengetahui seberapa sering model mengklasifikasikan data dengan benar [43]. Rumus akurasi ditunjukkan sebagai berikut:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \tag{2.6}$$

2.8.2 Precision

Pada kasus klasifikasi dengan jumlah kelas lebih dari dua, nilai akurasi belum dapat merepresentasikan performa model pada setiap kelas secara rinci. Oleh sebab itu, metrik *precision* dianggap lebih tepat untuk digunakan. Precision mengukur rasio antara jumlah prediksi positif yang benar terhadap total seluruh data yang diprediksi sebagai kelas positif [43]:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.7}$$

2.8.3 Recall

Recall mengukur sejauh mana model mampu mengenali kasus positif secara benar. Dengan demikian, recall menunjukkan kemampuan model dalam mengidentifikasi data positif aktual [43]:

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.8}$$

2.8.4 F1-score

F1-score merupakan nilai rata-rata harmonik yang diperoleh dari metrik *precision* dan *recall*. Metrik ini memperhitungkan keberadaan kesalahan *False Negative* (FN) maupun *False Positive* (FP), sehingga lebih tepat digunakan pada data dengan distribusi kelas yang tidak seimbang. Nilai F1-score maksimum

ketika precision dan recall memiliki nilai yang sama. Metrik ini juga memberikan keseimbangan terhadap kesalahan Tipe I dan Tipe II [43]:

$$F1\text{-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.9}$$

2.9 SUS (System Usability Scale)

SUS *(System Usability Scale)* merupakan alat ukur standar yang dikembangkan oleh John Brooke pada tahun 1986 untuk mengevaluasi tingkat *usability* atau kegunaan suatu sistem berdasarkan persepsi pengguna [44]. Instrumen ini berbentuk kuesioner yang terdiri dari sepuluh pernyataan, dengan skema penilaian menggunakan skala Likert 1 hingga 5, di mana nilai 1 menunjukkan "sangat tidak setuju" dan nilai 5 menunjukkan "sangat setuju". SUS banyak diterapkan karena mampu memberikan evaluasi yang cepat, mudah digunakan, serta memiliki tingkat keandalan yang baik dalam menilai pengalaman pengguna terhadap produk digital, seperti perangkat lunak, aplikasi, dan situs web [45].

Kerangka utama dari SUS berfokus pada pengukuran *perceived usability*, yaitu persepsi subjektif pengguna terhadap kemudahan penggunaan sistem secara keseluruhan. Penelitian lanjutan oleh Lewis dan Sauro (2009) menunjukkan bahwa SUS terdiri atas dua dimensi, yaitu *usability* (kegunaan umum, mencakup item 1, 2, 3, 5, 6, 7, 8, dan 9) serta *learnability* (kemudahan mempelajari sistem, mencakup item 4 dan 10). Dengan demikian, SUS tidak hanya menilai seberapa mudah sistem digunakan, tetapi juga seberapa cepat pengguna dapat memahaminya [46].

Struktur pernyataan pada SUS disusun secara bergantian antara pernyataan positif (ganjil) dan negatif (genap). Tujuan penyusunan ini adalah untuk mengurangi bias jawaban akibat kecenderungan responden memilih jawaban yang sama *(acquiescence bias)* tanpa memperhatikan isi pernyataan [46]. Misalnya, pernyataan nomor 1 seperti 'Saya ingin menggunakan sistem ini secara rutin' bersifat positif untuk menilai penerimaan pengguna terhadap sistem, sedangkan pernyataan nomor 2 seperti 'Sistem ini terlalu rumit untuk digunakan' bersifat negatif untuk menilai persepsi terhadap kompleksitas sistem.

2.9.1 Cara Perhitungan Skor SUS

Menurut Brooke (1996), proses penghitungan skor SUS dilakukan melalui beberapa langkah sebagai berikut [44]:

1. Setiap *item* dinilai menggunakan skala Likert dengan rentang nilai 1 hingga 5, di mana 1 merepresentasikan "sangat tidak setuju" dan 5 merepresentasikan "sangat setuju".
2. Untuk *item* dengan nomor ganjil (1, 3, 5, 7, dan 9), nilai kontribusi dihitung berdasarkan Persamaan 2.10 berikut:

$$\text{Nilai kontribusi} = (\text{Jawaban} - 1) \tag{2.10}$$

3. Untuk *item* dengan nomor genap (2, 4, 6, 8, dan 10), nilai kontribusi ditentukan menggunakan Persamaan 2.11:

$$\text{Nilai kontribusi} = (5 - \text{Jawaban}) \tag{2.11}$$

4. Seluruh nilai kontribusi dari sepuluh *item* kemudian dijumlahkan untuk memperoleh skor mentah dengan kisaran nilai antara 0 hingga 40.
5. Skor mentah tersebut selanjutnya dikalikan dengan faktor 2,5 untuk mendapatkan skor akhir SUS, sebagaimana ditunjukkan pada Persamaan 2.12:

$$\text{Skor SUS} = (\text{Total Nilai Kontribusi}) \times 2.5 \tag{2.12}$$

Hasil akhir dari SUS kemudian diinterpretasikan dengan acuan panduan yang dikembangkan oleh Bangor, Kortum, dan Miller (2009) serta Lewis dan Sauro (2018). Rata-rata skor SUS global adalah sekitar 68, yang dianggap sebagai ambang batas untuk kegunaan yang "cukup baik". Skor di atas 68 menunjukkan bahwa sistem memiliki tingkat kegunaan yang baik *(above average usability)*, sedangkan skor di bawah 68 menandakan adanya kelemahan dalam aspek kegunaan

(below average usability). Skor yang melebihi 80 dikategorikan memiliki kualitas kegunaan sangat baik *(excellent usability)*, sementara skor di bawah 51 masuk kategori rendah *(poor usability)* [45] [46] [47].

Data hasil evaluasi SUS dapat dikelompokkan berdasarkan kategori nilai, yaitu *Best imaginable, Excellent, Good, OK, Poor,* dan *Worst imaginable*. Semakin tinggi persentase skor pada kategori *Best imaginable, Excellent,* atau *Good*, semakin besar pula kepuasan dan penerimaan pengguna terhadap sistem. Sebaliknya, persentase tinggi pada kategori *Poor* atau *Worst imaginable* menunjukkan bahwa sistem masih memerlukan perbaikan, baik pada antarmuka, navigasi, maupun fungsionalitasnya [45].

2.10 Skala Likert

Skala Likert, yang diperkenalkan oleh Rensis Likert, merupakan alat ukur yang digunakan untuk menilai sikap seseorang. Pengukuran dilakukan dengan meminta responden menyatakan sejauh mana mereka setuju atau tidak setuju terhadap sejumlah pernyataan yang berkaitan dengan suatu topik tertentu [48]. Skala Likert lima tingkat adalah bentuk yang paling umum digunakan dalam penelitian karena dianggap memudahkan responden dalam memilih jawaban. Skala ini menyediakan lima opsi jawaban, yaitu Sangat Setuju, Setuju, Netral, Tidak Setuju, dan Sangat Tidak Setuju. Dalam penerapannya, Skala Likert mencakup dua jenis pernyataan, yakni positif dan negatif. Pernyataan positif diberikan skor 5, 4, 3, 2, dan 1, sedangkan pernyataan negatif dinilai dengan urutan skor terbalik, yaitu 1, 2, 3, 4, dan 5 [49].