BAB II

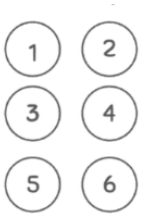
LANDASAN TEORI

2.1. Dasar Teori

Dalam subbab ini, akan dijelaskan beberapa dasar teori yang relevan dengan penelitian ini. Penjelasan mengenai teori-teori yang mendasari penelitian ini disampaikan dalam subbab berikutnya.

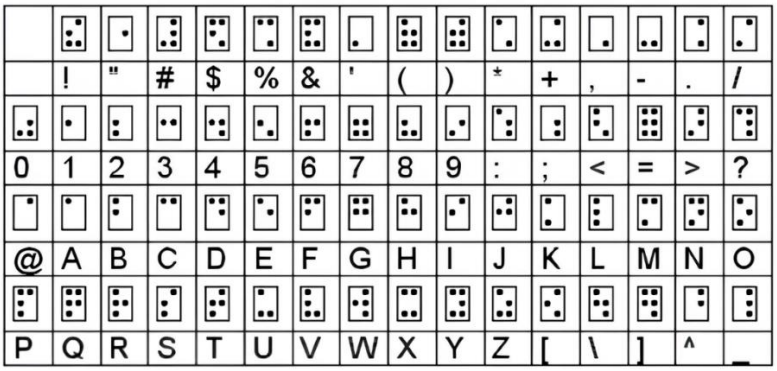
2.1.1 Huruf Braille

Menurut (Repelino et al., 2023) Braille adalah sistem pembacaan dan tulisan yang dikhususkan untuk tunanetra atau orang dengan gangguan penglihatan yang berat. Sel braille adalah komponen utama sistem tulisan ini, dan terdiri dari enam titik timbul per sel, masing-masing terdiri dari tiga baris dengan dua titik. Keenam titik ini dapat disusun sedemikian rupa sehingga menghasilkan berbagai kombinasi (Heni Herlina, 2022). Sebagai ilustrasi, Gambar 2.1 menunjukkan contoh titik-titik pada sel Braille yang dapat membentuk pola tertentu.



Gambar 2.1 Sel Braille (Heni Herlina, 2022)

Setiap karakter memiliki kombinasi yang berbeda, sehingga dengan enam titik ini, dapat dibuat sebanyak 64 kombinasi. Gambar 2.2 menunjukkan susunan titik-titik yang dikombinasikan pada huruf braille yang menunjukkan huruf abjad.



Gambar 2.2 Susunan kombinasi titik-titik huruf braille (Hamid Saifullah et al., 2019)

2.1.2 Pengolahan Citra Digital

Pemrosesan citra atau gambar merupakan langkah penting dalam analisis dan pengenalan gambar. Teknik pemrosesan gambar digunakan untuk meningkatkan kualitas gambar, mengurangi *noise*, dan mengekstrak informasi yang relevan untuk analisis lebih lanjut. beberapa teknik dasar pengolahan citra yang digunakan dalam penelitian ini akan disampaikan dalam subbab berikutnya.

2.1.2.1 Grayscale

Grayscale merupakan Metode pemrosesan citra mengubah gambar berwarna menjadi gambar hitam-putih (Teresa, 2019). Nilai intensitas setiap piksel dalam gambar *grayscale* berkisar mulai dari 0 (hitam) hingga 255 (putih). Mengubah gambar berwarna menjadi *grayscale* memudahkan analisis dan pemrosesan karena mengurangi kompleksitas data. Karena gambar *grayscale* lebih mudah dan lebih cepat diproses dibandingkan dengan gambar berwarna, gambar *grayscale* biasanya digunakan sebagai langkah pertama dalam pemrosesan citra (Mulyana, 2014).

Gambar 2.3 Grayscale level

Gambar 2.3 menunjukkan tingkatan *grayscale* dari warna putih ke hitam, yang digunakan untuk mengukur intensitas atau kecerahan dalam citra digital. Terdapat dua persamaan yang dapat digunakan untuk konversi ini. Persamaan (1) merata-ratakan nilai merah, hijau, dan biru, namun menghasilkan gambar yang kurang baik karena mata manusia tidak melihat ketiga warna tersebut secara sama. Oleh karena itu, persamaan (2) lebih sering digunakan, karena telah disesuaikan dengan sensitivitas mata manusia terhadap warna.

$$Y = \frac{R + G + B}{3} \tag{2.1}$$

$$\mathrm{Y} = 0.299\mathrm{R} + 0.587\mathrm{G} + 0.144B \tag{2.2}$$

Persamaan diatas merupakan formula yang digunakan untuk mengonversi gambar berwarna menjadi gambar *grayscale*.

Dengan keterangan:

R (*Red*) untuk Nilai intensitas merah pada piksel.

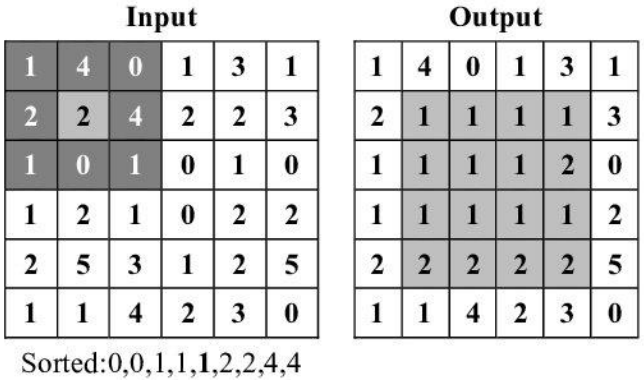
G (*Green*) untuk Nilai intensitas hijau pada piksel.

B (*Blue*) untuk Nilai intensitas biru pada piksel.

2.1.3 Median filter

Teknik pemrosesan citra yang dikenal sebagai *median filter* berfungsi untuk mengurangi *noise* dalam gambar (Wedianto et al., 2016). Nilai median dari piksel-piksel di sekitarnya dalam jendela *filter* yang telah ditentukan ditukar dengan nilai setiap piksel. *Median filter* menghilangkan *noise* sambil mengaburkan tepi gambar,

berbeda dengan *filter* rata-rata (Eliyani & Maulana, 2020). *Median filter* sangat bermanfaat untuk menghilangkan *noise* impulsif *salf and pepper* yang sering muncul dalam gambar digital.

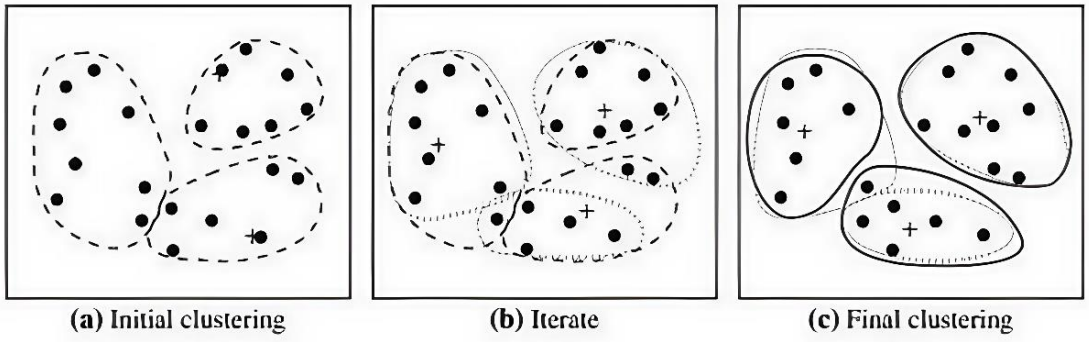


Gambar 2.4 Median filtering

Pada gambar 2.4 menunjukkan hasil dari penerapan *median filter* pada citra *input*, di mana setiap nilai di area 3x3 dari citra *input* diganti dengan nilai median dari area tersebut di citra *output*.

2.1.3 K-Mean Clustering

K-Means Clustering adalah teknik mengelompokkan data yang membagi kumpulan data menjadi k kelompok (*cluster*) (Prabowo et al., 2023). Tujuan utama dari teknik ini adalah untuk memaksimalkan jumlah kesamaan atribut dalam satu kelompok daripada jumlah yang ada di antara kelompok tersebut. Proses *K-Means Clustering* digambarkan pada Gambar 2.5.



Gambar 2.5 Ilustrasi Proses K-Means Clustering (Prabowo et al., 2023)

Proses *K-Means Clustering* digambarkan di bawah ini. Pada panel (a) terlihat pembagian awal data ke dalam beberapa *cluster* dengan *centroid* awal yang dipilih secara acak (ditunjukkan dengan tanda +). Pada panel (b), algoritma mulai beriterasi, di mana setiap data dipetakan ke *centroid* terdekat dan kemudian diperbarui berdasarkan posisi rata-rata dalam *cluster*. Pada panel (c), iterasi berakhir dengan pembentukan *cluster* akhir, di mana setiap data telah ditetapkan secara permanen ke *centroid* t.

Berikut adalah langkah-langkah dalam proses *K-Means Clustering* (Prabowo et al., 2023):

a. Tentukan jumlah k *cluster*. Pilih k titik data secara acak sebagai *centroid* awal.
b. Setelah *centroid* awal ditentukan, hitung jarak setiap data ke *centroid* terdekat menggunakan rumus Jarak *Euclidean*:

$$d\,(x,y)\; =\; \sqrt{\sum_{i=1}^{n}(x_i\; -\; y_i)^2} \qquad (2.3)$$

Dengan keterangan:

d(x,y) = Jarak *Euclidean* antara titik x dan y.

x dan y = Dua titik dalam ruang n-dimensi yang jaraknya akan dihitung.

c. Anggota kelompok diidentifikasi berdasarkan jarak data terkecil ke *centroid*.

d. Selanjutnya menghitung *centroid* baru menggunakan rumus mencari *centroid*:

$$\text{centroid}_i = \frac{\sum_{i=1}^{n} x_j}{n} \qquad (2.4)$$

Dengan keterangan:

Centroid = *Centroid* untuk *cluster* ke-i.

n = Jumlah total titik data dalam *cluster* ke-i.

e. Ulangi langkah 2 hingga 4 sampai tidak ada lagi data yang berubah *cluster*.

2.1.4 Metode Elbow

Metode *Elbow* adalah teknik yang digunakan untuk menentukan jumlah *cluster* optimal Metode ini berfungsi untuk mengevaluasi *cluster* (Dewi & Pramita, 2019). Jumlah *cluster* dianggap optimal ketika grafik membentuk titik siku. Perbandingan dilakukan dengan menghitung *Sum of Square Error* (SSE) (Hartanti, 2020).

$$SSE = \sum_{k=1}^{k} \sum_{x \in C_k} \|x - \mu_k\|^2 \qquad (2.5)$$

Dengan keterangan:

k = banyaknya *cluster*

x = data pada masing-masing *cluster*

ck = *cluster* ke-k

μ*k* = *centroid cluster* ke-k

2.1.5 Machine Learning (ML)

Machine learning adalah kumpulan teknik yang digunakan untuk mengatasi dan memprediksi data yang besar dengan mewakili data menggunakan algoritma

pembelajaran.Istilah *machine learning* pertama kali didefinisikan oleh Arthur Samuel pada tahun 1959 (Dompeipen & Sompie, 2020). Menurutnya, *machine learning* adalah bidang ilmu komputer yang memungkinkan komputer untuk belajar dan mengetahui sesuatu tanpa pemrograman eksplisit. *machine learning* adalah metode komputasi yang berdasarkan pengalaman untuk meningkatkan performa atau membuat prediksi yang akurat, di mana pengalaman merujuk pada informasi sebelumnya yang digunakan sebagai data pembelajaran (Jérémy et al., 2020).

Dalam machine learning, terdapat beberapa skenario pembelajaran (Jérémy et al., 2020):

1. *Supervised Learning* : Menggunakan data pembelajaran berlabel untuk membuat prediksi.
2. *Unsupervised Learning* : Menggunakan data pembelajaran tidak berlabel untuk mengelompokkan data berdasarkan karakteristik yang ditemukan.
3. *Reinforcement Learning* : Fase pembelajaran dan pengujian dicampur, di mana pembelajar mengumpulkan informasi dengan berinteraksi dengan lingkungan dan menerima balasan untuk setiap aksi.

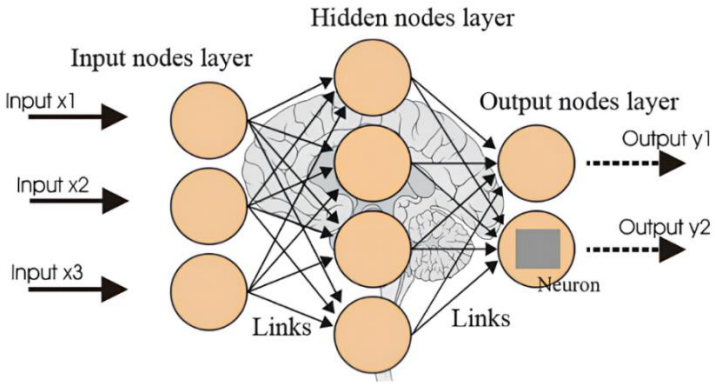
2.1.4 Deep Learning

Deep Learning adalah cabang dari *machine learning* yang menggunakan jaringan saraf tiruan atau dapat dianggap sebagai kemajuan dari jaringan saraf tiruan (Nurhakiki et al., 2024). Dengan menambahkan lebih banyak lapisan, model dapat lebih baik mewakili data berlabel. Sementara *machine learning* tradisional menggunakan ekstraksi fitur dan algoritma khusus untuk klasifikasi gambar dan pengenalan suara, metode ini seringkali kurang dalam hal kecepatan dan akurasi.

Namun, *deep learning*, melalui jaringan syaraf tiruan yang dalam, memungkinkan pembelajaran dengan kecepatan, akurasi, dan skala yang lebih besar, menjadi lebih sering digunakan dalam riset dan industri untuk memecahkan masalah data besar seperti *computer vision* (Soekarta et al., 2023).

2.1.5 Artificial Neural Network

Neural Network atau sering juga disebut *Artificial Neural Networks (ANNs)* adalah bidang penelitian yang menarik dan dinamis dalam pembelajaran mesin dan kecerdasan buatan (Maad M. Mijwel, 2021).



Gambar 2.6 *Lapisan Artificial Neural Network* (Maad M. Mijwel, 2021)

Pada gambar 2.6 menunjukkan bahwa jaringan saraf terdiri dari tiga lapisan *input, Hidden,* dan *output.*

1) Lapisan masukan (*input*), terdiri dari sejumlah *neuron* yang akan menerima sinyal dari luar dan kemudian mengirimkannya ke lapisan lain dalam jaringan.
2) Lapisan tersembunyi (*hidden layer*), terletak di atas lapisan masukan dan lapisan keluaran, dan berfungsi untuk meningkatkan kemampuan jaringan untuk memecahkan masalah.

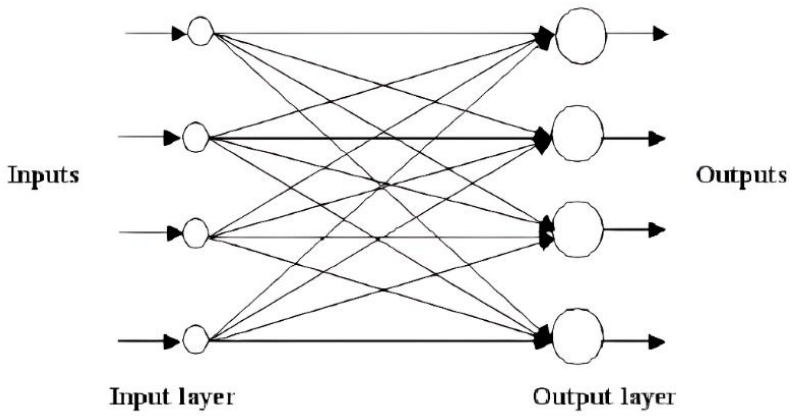
3) Lapisan keluaran (*output*), untuk mengirimkan sinyal yang dihasilkan dari pemrosesan jaringan

2.1.2.1 Arsitektur neural network

Beberapa arsitektur jaringan *Neural Network* adalah sebagai berikut:

1. *Single Layer Network*

Neuron pada lapisan *input* tidak memiliki fungsi aktivasi. Sebaliknya, *neuron* pada lapisan tersembunyi dan lapisan *output* kadang-kadang memiliki fungsi aktivasi yang berbeda tergantung pada data atau masalah yang dimiliki (Wijaya, 2019). Gambar 2.6 menunjukkan gambaran dari suatu sambungan satu lapisan.

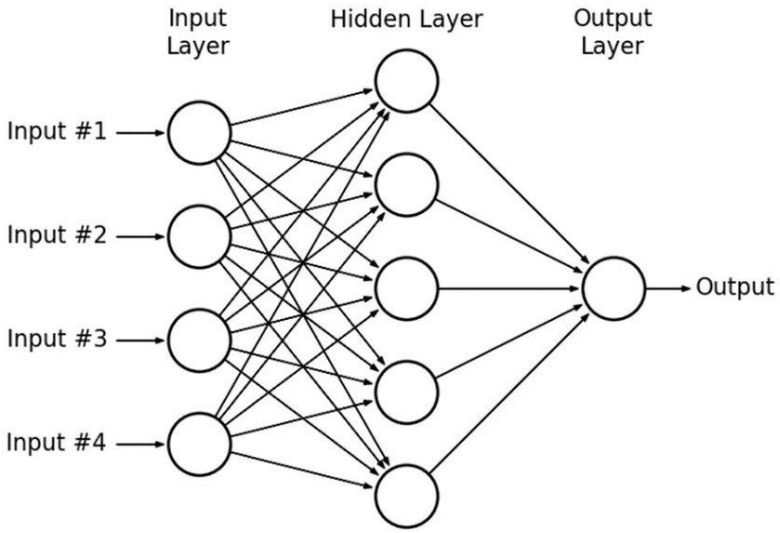


Gambar 2.7 *Single Layer Network* (Senok et al., 2018)

Gambar 2.7 menunjukkan arsitektur *Single Layer Network* dengan lapisan input, lapisan keluaran, dan koneksi penuh antara setiap *neuron* pada lapisan *input* ke setiap *neuron* pada lapisan *output*.

2. *Multi Layer Perseptron*

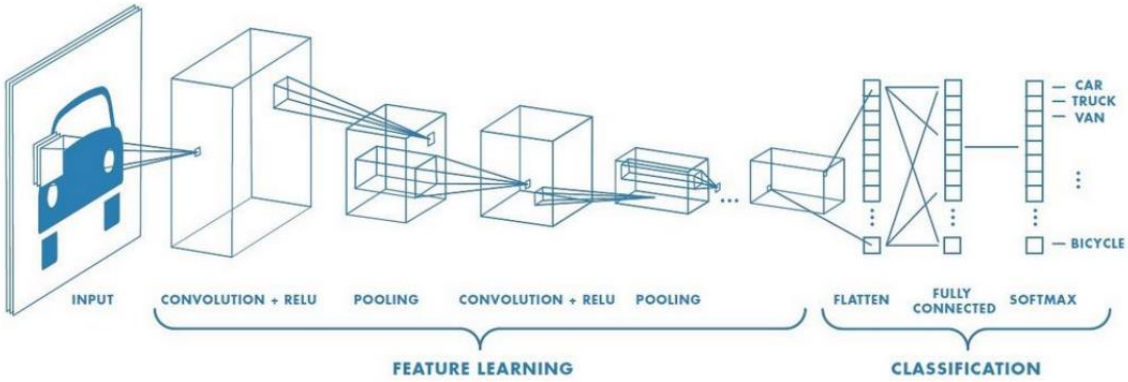
Multilayer Perceptron (MLP) adalah turunan dari *Perceptron* dan berupa *feedforward neural network* dengan satu atau lebih lapisan *hidden*. MLP biasanya terdiri dari satu lapisan *input*, setidaknya satu lapisan *hidden*, dan satu lapisan *output.* Per *layer*, sinyal *input* dirambatkan dengan arah maju dari *input* ke *output* (Wijaya, 2019).Gambar 2.8 menunjukan *Multi Layer Perseptron.*



Gambar 2.8 *Multilayer Layer Network* (Chinazzo, 2019)

2.1.6 Convolutional Neural Network (CNN)

Convolutional Neural Network adalah pengembangan dari *Multilayer Perceptron* yang dirancang untuk mengolah data dua dimensi seperti citra (Suartika E. P, I Wayan, Wijaya Arya Yudhi, 2016). CNN termasuk dalam kategori *Deep Neural Network* karena memiliki kedalaman jaringan yang tinggi dan sering diterapkan pada data citra.

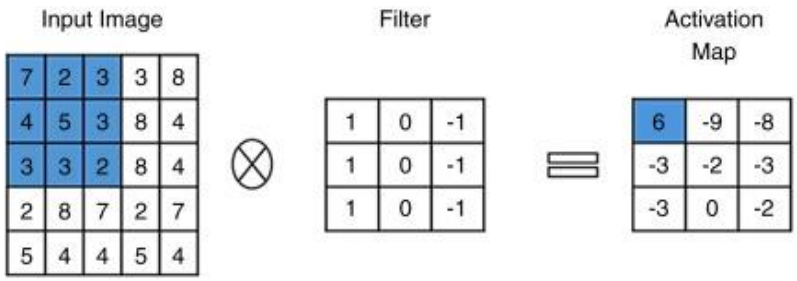


Gambar 2.9 Arsitektur *Convolutional Neural Network* (Prihatiningsih et al., 2019)

Berdasarkan gambar 2.9 tahap pertama dalam arsitektur *Convolutional Neural Network* adalah tahap konvolusi. Pada tahap ini, sebuah kernel dengan ukuran tertentu digunakan. Jumlah *kernel* yang dipakai bergantung pada jumlah fitur yang dihasilkan. Setelah itu, dilanjutkan dengan fungsi aktivasi, biasanya menggunakan fungsi ReLU (*Rectifier Linear Unit*). Setelah melewati proses fungsi aktivasi, dilanjutkan dengan proses *pooling*. Proses ini diulang beberapa kali hingga diperoleh peta fitur yang cukup untuk dilanjutkan *ke fully connected neural network*. *Output* dari *fully connected network* adalah *ouput class*.

2.1.7.1 Convolution Layer

Lapisan konvolusi (*Convolution Layer*)merupakan bagian penting dari arsitektur CNN, yang melakukan operasi konvolusi pada *output* dari lapisan sebelumnya (Suartika E. P, I Wayan, Wijaya Arya Yudhi, 2016). Lapisan ini adalah proses utama yang mendasari jaringan arsitektur CNN. Konvolusi adalah istilah matematis untuk penerapan fungsi secara berulang pada *output* fungsi lain. Operasi konvolusi melibatkan dua fungsi dengan argumen bernilai nyata, di mana hasilnya adalah peta fitur (*feature map*) dari citra *input.*



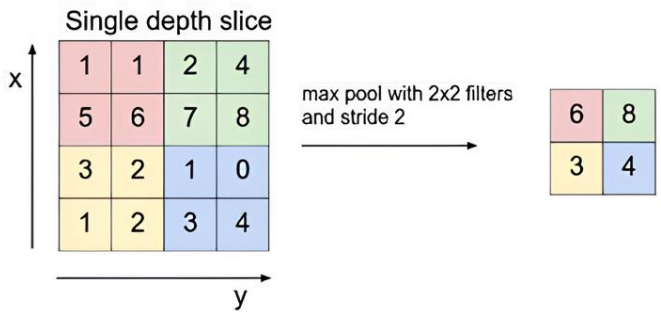
Gambar 2.10 Contoh Convolution Layer (Mostafa & Wu, 2021)

Dalam gambar 2.10, Convolutional Neural Network (CNN) mengonvolusikan gambar. Matriks angka 6x6 menunjukkan gambar *input* di sebelah kiri. Sebuah *filter* (*kernel*) berukuran 3 kali 3 berada di tengah dan memiliki nilai tertentu. Proses konvolusi dilakukan dengan menggeser *filter* di atas gambar input. Untuk melakukan ini, setiap posisi menghitung hasil perkaliannya dengan bagian yang sesuai dari gambar *input* dan kemudian menjumlahkan hasil perkalian tersebut untuk menghasilkan nilai pada peta aktivasi (*Activation Map*) di sebelah kanan. Sebagai contoh, bagian pertama gambar *input* (dalam warna biru) dikalikan dengan *filter*, dan hasilnya adalah 6.

2.1.7.2 Operasi Pooling

Pooling adalah proses pengurangan ukuran matriks dengan menggunakan operasi *pooling*, biasanya dilakukan setelah lapisan konvolusi (Andreas Restu Priatama, 2023). *Pooling layer* terdiri dari *filter* dengan ukuran dan *stride* tertentu yang bergerak melintasi seluruh *area feature map*. Ada dua jenis *pooling* yang umum digunakan: *average pooling*, yang mengambil nilai rata-rata, dan *max-pooling*, yang mengambil nilai maksimal. *Pooling layer* yang ditempatkan di antara lapisan konvolusi dalam arsitektur CNN secara bertahap mengurangi ukuran *volume output* pada *feature map*, sehingga mengurangi jumlah parameter dan

perhitungan dalam jaringan untuk mengendalikan *overfitting*. *Pooling layer* bekerja pada setiap tumpukan *feature map* dan mengurangi ukurannya. Biasanya, *pooling layer* menggunakan *filter* berukuran 2x2 dengan langkah dua yang diaplikasikan pada setiap irisan input.



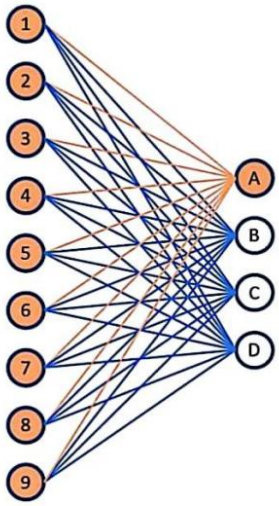
Gambar 2.11 Contoh Pooling (Muhammad Afif Amanullah Fawwaz, Kurniawan Nur Ramadhani, S.T. & Febryanti Sthevanie, 2021)

Gambar 2.11 menunjukkan proses *max-pooling*. *Output* dari proses ini adalah matriks dengan dimensi lebih kecil dibandingkan citra awal. *Pooling layer* beroperasi pada setiap irisan kedalaman *volume input* secara bergantian. Dalam contoh pada gambar, operasi *max-pooling* menggunakan *filter* berukuran 2x2. *Input* berukuran 4x4, dan dari setiap kelompok 4 angka pada *input*, nilai maksimal diambil untuk menghasilkan *output* baru berukuran 2x2.

2.1.7 Fully-Conected Layer

Fully-Connected Layer adalah lapisan di mana semua *neuron* dari lapisan sebelumnya terhubung dengan semua *neuron* di lapisan berikutnya, mirip dengan *neural network* biasa (Christianto et al., 2021). Lapisan ini umumnya digunakan dalam *Multi Layer Perceptron* (MLP) untuk mentransformasikan dimensi data sehingga dapat diklasifikasikan secara linier. Perbedaannya dengan lapisan

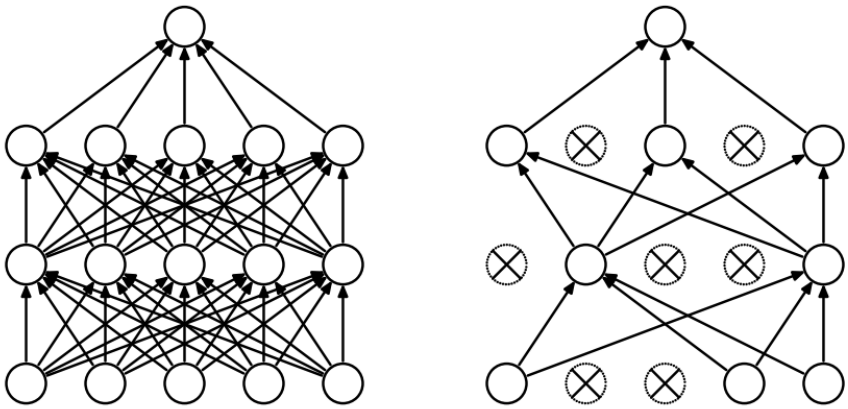
konvolusi adalah bahwa *neuron* di lapisan konvolusi hanya terhubung ke area tertentu pada *input*, sementara *neuron* di lapisan *fully-connected* terhubung secara keseluruhan. Meskipun demikian, keduanya menggunakan operasi *dot product*, sehingga fungsi dasarnya tidak jauh berbeda.



Gambar 2.12 Contoh Fully-Connected Layer (Ye, 2024)

2.1.7 Dropout Regularization

Dropout adalah teknik regularisasi dalam jaringan saraf yang bertujuan untuk menghapus sebagian *neuron* secara acak selama proses pelatihan. Dengan kata lain, beberapa *neuron* akan dinonaktifkan secara acak pada setiap iterasi pelatihan (Christianto et al., 2021). Hal ini berarti kontribusi *neuron* yang dinonaktifkan tidak diperhitungkan dalam jaringan saat itu, dan bobot baru tidak diterapkan pada *neuron* tersebut selama proses *backpropagation*. Teknik *dropout* membantu mencegah *overfitting* dengan memaksa jaringan untuk menjadi lebih *robust* dan tidak terlalu tergantung pada *neuron* tertentu.



Gambar 2.13 ilustrasi *dropout* (Ikasari et al., 2017)

Gambar 2.13 menunjukkan perbandingan antara jaringan saraf biasa yang di tunjukan Gambar 2.13 pada bagian a yang memiliki dua lapisan tersembunyi, dan jaringan saraf dengan *dropout*. Pada bagian b, teknik *dropout* diterapkan, yang menghasilkan beberapa *neuron* aktivasi yang tidak lagi digunakan. Ini bertujuan untuk mengurangi *overfitting* dengan secara acak mengabaikan sebagian *neuron* saat proses pelatihan, sehingga memaksa jaringan untuk belajar fitur yang lebih umum dan mencegah ketergantungan terlalu kuat pada *neuron* individu.

2.1.7 Softmax Layer

Softmax adalah fungsi aktivasi yang digunakan pada *layer output*. *Output* dari *softmax* digunakan untuk memprediksi probabilitas kelas pada masalah klasifikasi multikelas. Fungsi *softmax* umumnya digunakan dalam berbagai metode klasifikasi seperti *regresi logistik multinomial,* analisis diskriminan linear multikelas, *naive Bayes classifier*, dan dalam jaringan *neural.* Secara khusus, *softmax* sering digunakan dalam *regresi logistik multinomial* dan analisis

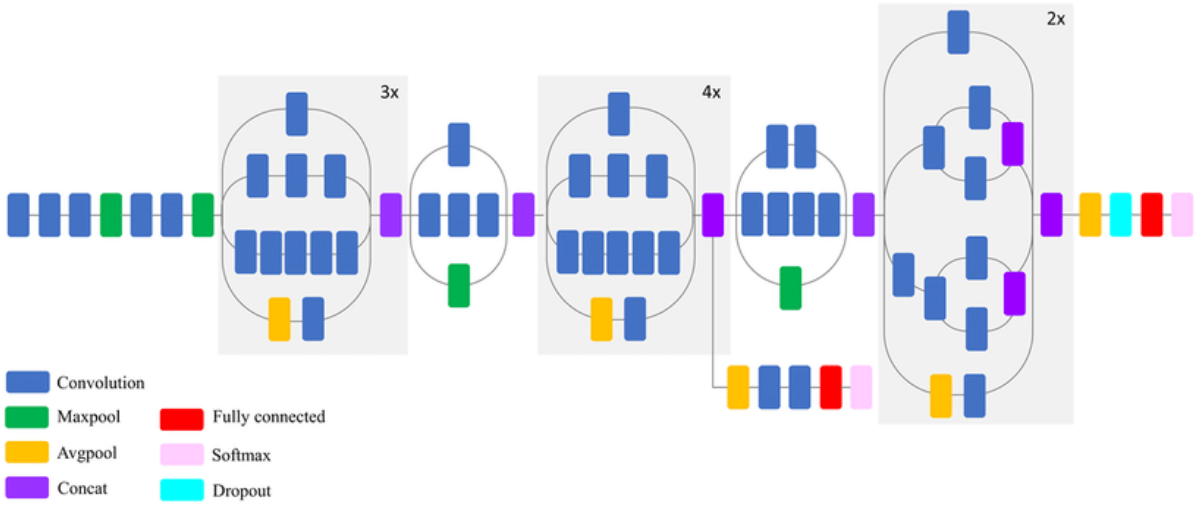
diskriminan *linear* multikelas untuk menghitung probabilitas kelas yang diinginkan.

$$f(x)= \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \quad (2.6)$$

Notasi (f_j) merujuk pada hasil fungsi untuk setiap elemen ke-j pada vektor keluaran kelas. Argumen (z) adalah hipotesis yang dihasilkan oleh model *training* untuk klasifikasi menggunakan fungsi *softmax*. Fungsi *softmax* memberikan hasil yang intuitif dan memiliki interpretasi probabilistik yang baik, dibandingkan dengan algoritma klasifikasi lainnya. *Softmax* memungkinkan perhitungan probabilitas untuk semua label kelas. Ini mengambil vektor nilai riil dan mentransformasinya menjadi *vektor* dengan nilai antara nol dan satu, di mana jumlah semua nilai dalam *vektor* tersebut adalah satu.

2.1.7 Inception V3

Inception V3 merupakan arsitektur CNN yang mampu melakukan pengenalan gambar, termasuk klasifikasi gambar (Ungkawa & Hakim, 2023). *Inception V3* adalah salah satu arsitektur CNN yang dikembangkan dari versi sebelumnya, yaitu *GoogLeNet* (*Inception*) dan *Inception V2*. *GoogLeNet* pertama kali diperkenalkan pada tahun 2014 dan dinobatkan sebagai arsitektur terbaik dalam kompetisi *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) 2014. Inception memperkenalkan teknik baru dalam arsitektur CNN dengan memfaktorkan *convolution layer* menjadi *multi-layer* dengan ukuran *kernel* yang lebih kecil. Penggunaan teknik ini dapat mengurangi jumlah parameter dengan berbagi bobot dalam *multi-laye*r tersebut, serta memungkinkan pembuatan arsitektur yang lebih dalam tanpa menambah biaya komputasi (Huda et al., 2023).



Gambar 2.14 Arsitektur Inception V3 (Karpe et al., 2023)

2.1.8 Confusion matrix

Confusion matrix adalah alat evaluasi dalam *machine learning* yang digunakan untuk menganalisis kinerja model klasifikasi. Matriks ini menampilkan hasil prediksi model dalam bentuk tabel dengan empat komponen utama: *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). TP dan TN menunjukkan jumlah prediksi yang benar untuk kelas positif dan negatif, sementara FP dan FN menunjukkan jumlah kesalahan prediksi untuk masing-masing kelas. Dengan menggunakan *confusion matrix*, metrik evaluasi seperti akurasi, presisi, *recall*, dan *F1-score* dapat dihitung untuk menilai seberapa baik model dalam mengenali setiap kelas (Rahayu et al., 2021).

2.1.9 Recall, Precision, f1 Score dan Accuracy

Recall, Precision, F1 Score, dan Accuracy adalah metrik evaluasi yang sering digunakan untuk mengukur kinerja model dalam tugas klasifikasi, terutama ketika menangani data yang tidak seimbang (Iskandar & Salam, 2024). Recall

mengukur kemampuan model dalam mengenali semua *instance* dari kelas tertentu, menunjukkan sensitivitas model. *Precision* menilai ketepatan model dalam memprediksi kelas dengan benar, atau seberapa sering prediksi positif model benar-benar positif. *F1 Score* adalah rata-rata harmonis dari *Precision* dan *Recall*, memberikan gambaran yang seimbang antara keduanya. Nilai ini sangat berguna ketika ada *trade-off* antara *Precision* dan *Recall*. *Accuracy* menggambarkan proporsi total prediksi yang benar, baik untuk kelas positif maupun negatif, dari seluruh *dataset*. metrik-metrik ini memberikan gambaran lengkap tentang performa model, membantu menentukan seberapa baik model tersebut dalam klasifikasi data yang diuji. Untuk rumus dari matrik evaluasi terdapat pada persamaan 2.2 sampai persamaan 2.5 (Rahayu et al., 2021).

$$\text{Akurasi} = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.7}$$

$$\text{Presisi} = \frac{TP}{TP + FP} \tag{2.8}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.9}$$

$$F1\ Score = 2 \times \frac{Recall\ x\ Presisi}{Recall + Presisi} \tag{2.10}$$

Penjelasan pada persamaan TP (*True Positive),*TN *(True Negative),*FP *(False Positive),* dan FS *(False Negative).*

2.1.9 Early stopping

Early stopping adalah teknik regulasi dalam pembelajaran mesin yang digunakan untuk mencegah *overfitting* selama pelatihan model (Samidin & Fadjeri, 2024). Teknik ini memantau kinerja model pada data validasi selama proses pelatihan, dan menghentikan pelatihan ketika kinerja model mulai menurun atau

tidak meningkat lagi setelah sejumlah iterasi tertentu, meskipun pada data pelatihan masih ada peningkatan. Dengan demikian, *early stopping* membantu memastikan bahwa model tidak terlalu beradaptasi terhadap data pelatihan dan lebih mampu melakukan generalisasi pada data yang belum pernah dilihat.

2.1.10 Learning Rate

Learning rate adalah parameter yang mengatur seberapa cepat algoritma memperbarui bobot jaringan, dengan mempertimbangkan penurunan nilai *error* (Dananjaya et al., 2022). *Learning rate* mengontrol seberapa besar perubahan yang dilakukan pada bobot model pada setiap literasi pembaruan. Jika terlalu tinggi, model bisa melewatkan solusi ideal karena pembaruan bobot yang terlalu besar, menyebabkan model tidak konvergen atau tidak stabil. Sebaliknya, jika terlalu rendah, proses pelatihan bisa sangat lambat dan mungkin terjebak pada minimum lokal.

2.1.11 Batch Size

Batch Size adalah jumlah sampel data yang disebarkan ke *Neural Network* (Azmi et al., 2023). Data dipecah menjadi beberapa *batch* yang lebih kecil daripada memproses seluruh *dataset* sekaligus, yang membutuhkan banyak memori dan memakan waktu. Setiap *batch* melewati passing ke depan, tempat prediksi dibuat, dan passing ke belakang, tempat kesalahan dihitung dan bobot model diperbarui. Memanfaatkan *batch* memungkinkan model untuk belajar dengan lebih efisien dan mempercepat proses pelatihan sambil membantu mengurangi variasi yang signifikan dalam pembaruan parameter model.

2.1.12 Epoch

Epoch adalah satu siklus lengkap di mana seluruh dataset digunakan untuk melatih *Neural Network* hingga kembali ke awal untuk satu putaran (Azmi et al., 2023). Karena satu *epoch* sering kali terlalu besar untuk dimasukkan ke dalam komputer sekaligus, *dataset* dibagi menjadi bagian-bagian kecil yang disebut *batches.* Dalam setiap *epoch*, bobot/*weight* jaringan diperbarui.

2.2 Penelitian Terkait

Tabel 2.1 Penelitian Terkait

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model /	Hasil
1	(Ronando & Sudaryanto, 2018)	Sistem Pengenalan Pola Huruf Braille Berbasis *Audio* Menggunakan Metode *Naïve Bayes*	*Naive Bayes*	Hasil evaluasi menunjukkan bahwa sistem ini dapat mengenali huruf braille dengan akurasi 88,172% dan membutuhkan waktu rata-rata 5 *sekon* untuk merespons *file audio*.kekurangan dari penelitian ini Kekurangan sistem adalah waktu respon yang relatif lambat dalam mengubah hasil pengenalan pola huruf braille menjadi *audio*, dengan rata-rata 5 detik.
2	(Wahyu et al., 2024)	Pengenalan Pola Aksara Batak menggunakan *Backpropagation*	*artificial neural network*	hasil dari pengujian metode *Bacpropagation* menunjukkan bahwa pengenalan mencapai 97 persen dari data gambar yang diidentifikasi.
3	(Meena et al., 2023)	*Sentiment analysis on images using convolutional neural networks based Inception-V3 transfer learning approach*	*Convolutional Neural Network*	Penelitian menunjukkan bahwa metode yang diusulkan dapat mencapai tingkat akurasi 99,5%.

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model /	Hasil
4	(Riva et al., 2022)	Perbandingan Algoritma Cnn Dan Ann Dengan *Projection* Histogram Untuk Klasifikasi Citra Tulisan Tangan Berupa Angka	*Artificial Neural Network* dan *Convolutional Neural Network*	Percobaan menunjukkan bahwa algoritma CNN memiliki akurasi 0,9752 pada data validasi dan 0,9689 pada data tes. Sebaliknya, algoritma *projection histogram ANN* memiliki akurasi 0,8904 pada data validasi dan 0,8962.
5	(Jahandad et al., 2019)	*Offline Signature Verification using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3*	*Convolutional neural network*	arsitektur *GoogLeNet* CNN yang populer, yaitu, *Inception-v1* dan *Inception-v3*, digunakan. Pertama, algoritma dilatih pada sampel dari 20 pengguna, dan mencapai akurasi validasi sebesar 83% untuk *Inception-v1* dan 75% untuk *Inception-v3*.
6	(Ahmed et al., 2023)	*An inception V3 approach for malware classification using machine learning and transfer learning*	*Convolutional neural network* dan LSTM	Pendekatan pembelajaran *transfer* menggunakan *InceptionV3* menghasilkan kinerja yang baik sehubungan dengan model yang dibandingkan seperti LSTM dengan akurasi klasifikasi 98,76% pada kumpulan data uji dan 99,6% pada kumpulan data latih.

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model /	Hasil
7	(Andriansyah, Marindo, 2021)	Pengenalan Karakter Braille Memanfaatkan *Convolutional Neural Network*	*Convolutional Neural Network*	Hasil penelitian menunjukkan CNN memproses 3 kelompok data, 60, 100, dan 150 data dengan masing-masing data menggunakan 5, 10, 25, dan 50 *epoch*. Nilai akurasi tertinggi pada proses *traning* sebesar 99.87% dengan nilai *loss* sebesar 0.232. Dalam proses pengenalan akurasi tertinggi sebesar 99.62%
8	(Neshat et al., 2024)	*Hybrid Inception Architecture with Residual Connection: Fine-tuned Inception-ResNet Deep Learning Model for Lung Inflammation Diagnosis from Chest Radiographs*	*Convolutional Neural Network*	akurasi klasifikasi, *Inception-ResNet-V2* lebih unggul dibandingkan dengan model lain, termasuk *ResNet152V2*, *MobileNet-V3* (Besar dan Kecil), *EfficientNetV2* (Besar dan Kecil), *InceptionV3*, dan *NASNet-Mobile*, dengan *margin* yang substansial. Ia mengungguli mereka masing-masing sebesar 2,6%, 6,5%, 7,1%, 13%, 16,1%, 3,9%, dan 1,6%,
9	(Thangaraj et al., 2023)	*A deep convolution neural network for automated COVID-19 disease detection using chest X-ray images*	*Convolutional Neural Network*	Hasil eksperimen menunjukkan bahwa model *MIn-V3* melampaui model lain yang telah dilatih sebelumnya dengan akurasi klasifikasi sebesar 96,33%.

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model /	Hasil
10	(Grijalva et al., 2023)	*Image classification of sugarcane aphid density using deep convolutional neural networks*	*Convolutional Neural Network*	Arsitektur *Inception v3* dan *DenseNet* masing-masing mencapai akurasi 83% dan 81%
11	(Korium et al., 2024)	*Image-based intrusion detection system for GPS spoofing cyberattacks in unmanned aerial vehicles*	*Convolutional Neural Network*	Namun, kinerja *Inception-v3* meningkat secara signifikan setelah pengoptimalan Bayesian, dengan *Tree-structured Parzen Estimator* mencapai akurasi 99,06%.
12	(Raghuwanshi et al., 2024)	*Early Detection of Brain Tumor from MRI Images Using Different Machine Learning Techniques*	*Regresi Logistik*, KNN, VGG19, *Inception V3* dan VGG19	Akurasi model yang diusulkan untuk *Regresi Logistik*, KNN, VGG19, *Inception V3* dan VGG19 dengan augmentasi masing-masing adalah 84,66%, 90,68%, 92,17%, 91,56% dan 95,43%.
13	(WU et al., 2020)	*Detection and enumeration of wheat grains based on a deep learning method under various scenarios and scales*	*Convolutional Neural Network*	Akurasi model mencapai presisi rata-rata 0,91 dengan total kerugian model di bawah 0,5

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model /	Hasil
14	(Al-Salman & AlSalman, 2024)	Fly-LeNet: *A deep learning-based framework for converting multilingual braille images*	*Convolutional Neural Network*	Model yang disarankan mencapai akurasi klasifikasi tinggi sebesar 99,77% dan 99,80% pada set uji kumpulan data pertama dan kedua.
15	(Elaraby et al., 2024)	*A generalized ensemble approach based on transfer learning for Braille character recognition*	*Convolutional Neural Network*	nilai lebih besar daripada masing-masing model pembelajaran *transfer* individu. Skor F1 dari *ensemble* yang diperkenalkan mencapai 89,42%, 99,58%, dan 97,11%
16	(Gadekallu et al., 2022)	*Hand gesture recognition based on a Harris Hawks optimized Convolution Neural Network*	*Convolutional Neural Network*	NN yang diusulkan mengungguli model yang ada dengan mencapai Akurasi 100%
17	(Fathur Rozi et al., 2023)	Identifikasi Kinerja Arsitektur *Transfer Learning* Vgg16, *Resnet-50*, Dan *Inception-V3* Dalam Pengklasifikasian Citra Penyakit Daun Tomat	*Convolutional Neural Network*	arsitektur *Inception-V3*, nilai akurasi dan validasi akurasi masing-masing 0,9551 dan 0,9544, sedangkan untuk arsitektur *ResNet50*, masing-masing 0,9578 dan 0,9467. Dengan arsitektur VGG16, nilai akurasi tertinggi adalah 0,9754 dan validasi akurasi tertinggi adalah 0,9778.

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model	Hasil
18	(Udapola & Liyanage, 2017)	*Braille Messenger: Adaptive Learning Based Non- Visual Touch Screen Text Input for the Blind Community Using Braille*	K-NN dan K-Means	Akurasi deteksi karakter Braille sebesar 97.4% dan akurasi deteksi pola gambar sebesar 94.86% menunjukkan bahwa aplikasi juga dapat mengenali karakter Braille yang diketik oleh pengguna dengan tingkat keakuratan yang tinggi.
19	(Nugraha et al., 2022)	Deteksi Objek Dan Jenis Burung Menggunakan *Convolutional Neural Network* Dengan Arsitektur *Inception Resnet-V2*	*Convolutional Neural Network*	Hasil didapat ketika gausian noise ditambahkan dan citra diubah menjadi *grayscale* pada setiap gambar. Dengan menggunakan optimaliasi *Adamax*, akurasi diperoleh sebesar 93.33% dengan *learning rate* 0.001, kemudian meningkat menjadi 95.98% dengan *learning rate* 0.0001. Pada pengujian yang tidak memberikan *grayscale* dan *gausian noise*, akurasi tertinggi diperoleh sebesar 98.28% dengan *learning rate* 0.001 0.001, kemudian dengan *learning rate* 0.001 0.001.

No	Peneliti / Tahun	Judul	Metode / Algoritma / Teknik / Model	Hasil
20	(Tang et al., 2018)	*Median filtering detection of small-size image based on CNN*	*Convolutional Neural Network*	penelitian ini menunjukkan bahwa *MFNet* dapat secara efektif mendeteksi penyaringan median pada gambar kecil dan terkompresi, meningkatkan akurasi dan kinerja deteksi secara keseluruhan.
21	(Masa & Hamdani, 2021)	Klasifikasi Motif Citra Batik Menggunakan *Convolutional Neural Network* Berdasarkan *K-means Clustering*	*Convolutional Neural Network*	Hasil penelitian menunjukkan bahwa klasifikasi citra batik dengan CNN berdasarkan *K-means Clustering* dari hasil *median filter* mencapai akurasi 100%, sedangkan dari hasil *sharpening* memperoleh akurasi 80%.
22	(Bayu Adhya Wiratama, 2024)	Sistem Pengenalan Huruf Braille Menggunakan Metode *Deep Learning* Berbasis Website	*Convolutional Neural Network*	Beberapa metode deep learning, seperti Base Convolutional Neural Network (CNN), VGG-16, ResNet50, dan Inception-V3, diusulkan dalam penelitian ini. Dataset 12.641 gambar karakter Braille yang digunakan berasal dari AEyeAlliance, yang dibagi menjadi 37 kelas. Model Base CNN menunjukkan akurasi

				pelatihan sebesar 98%, akurasi validasi sebesar 99%, dan akurasi pengujian sebesar 99,1%.

2.3 Matrik Penelitian

Matrik penelitian menunjukkan berbagai metode untuk memecahkan masalah. Ini juga menjelaskan bagaimana penelitian sebelumnya berhubungan dengan penelitian saat ini. Tabel 2.2 berisi matrik penelitian

Tabel 2.2 Matrik Penelitian

No	Peneliti	CNN	Inception-V3	Segmentasi	Median filtering
1.	(Ronando & Sudaryanto, 2018)			✓	✓
2.	(Wahyu et al., 2024)				✓
3.	(Meena et al., 2023)	✓	✓		
4.	(Laurenza Setiana Riva, Felicia Febriana, Muthiara Panghurina, 2022)	✓			
5.	(Jahandad et al., 2019)	✓			
6.	(Ahmed et al., 2023)	✓	✓		
7.	(Andriansyah, Marindo, 2021)	✓		✓	✓
8.	(Neshat et al., 2024)	✓	✓	✓	
9.	(Thangaraj et al., 2023)				
10.	(Grijalva et al., 2023)	✓	✓		
11.	(Korium et al., 2024)	✓	✓		
12.	(Raghuwanshi et al., 2024)	✓	✓	✓	
13.	(WU et al., 2020)	✓		✓	

14.	(Al-Salman & AlSalman, 2024)	✓		✓	
15.	(Elaraby et al., 2024)	✓			

No	Peneliti	CNN	Inception-V3	Segmentasi	Median filtering
16.	(Gadekallu et al., 2022)	✓			
17.	(Fathur Rozi et al., 2023)	✓	✓		
18.	(Udapola & Liyanage, 2017)				
19.	(Nugraha et al., 2022)	✓		✓	
20.	(Tang et al., 2018)	✓			✓
21.	(Masa & Hamdani, 2021)	✓		✓	✓
22.	(Bayu Adhya Wiratama, 2024)	✓	✓	✓	
23.	(Latip, 2024)	✓	✓	✓	✓

Berdasakan 22 penelitian terkait yang dijadikan referensi, dengan 5 di antaranya menjadi dasar penelitian ini. (Andriansyah, Marindo, 2021; Bayu Adhya Wiratama, 2024; Herlambang et al., 2021; Ronando & Sudaryanto, 2018), penelitian akan mengabungkan metode atau algoritma yang digunakan untuk menjadi rujukan penelitian sebelumnya, Penelitian ini akan mengembangkan model deteksi pola Braille menggunakan arsitektur *Inception V3* dengan mengunakan *median filter* dan segmentasi. Penelitian sebelumnya telah menunjukkan potensi CNN dalam pengenalan Braille dengan akurasi yang bervariasi, seperti 81,54% (Herlambang et

al., 2021). *Inception V3* terbukti efektif dengan akurasi 95,5% (Bayu Adhya Wiratama, 2024). Namun, teknik praproses seperti *median filtering* dan segmentasi belum sepenuhnya dioptimalkan,pada arsitektur *inception V3* sehingga penelitian ini akan berfokus pada akurasi melalui integrasi kedua teknik tersebut.