

BAB I

PENDAHULUAN

1.1 Latar Belakang

Seiring berkembangnya industri pengolahan makanan, permintaan jagung juga meningkat. Jagung merupakan makanan pokok kedua setelah beras [1]. Tanaman jagung tumbuh baik di daerah panas dan dingin dengan curah hujan dan irigasi yang cukup. Terjadinya penyakit tanaman memiliki efek negatif pada produksi pertanian dan jika penyakit tanaman tidak terdeteksi pada waktunya, maka akan terjadi peningkatan kerawanan pangan [2]. Namun selama satu siklus hidupnya dari benih ke benih, setiap bagian jagung peka terhadap sejumlah penyakit terutama pada daun jagung.

Penyakit tanaman memiliki dampak buruk pada keamanan produksi pangan, dan dapat menyebabkan penurunan kualitas dan kuantitas produk pertanian. Dalam kasus yang parah, penyakit tanaman bahkan dapat menyebabkan tidak ada panen biji-bijian sama sekali [3]. Penurunan produksi jagung dapat diakibatkan oleh patogen *Helminthosporium turcicum* yang menyebabkan bercak daun pada jagung. Penyakit karat daun disebabkan oleh jamur *Puccinia sorghi*. Gejala awal berupa bercak-bercak merah dan keluar serbuk seperti tepung berwarna coklat kekuningan. Akibat penyakit ini, tanaman tidak dapat melakukan fotosintesis dengan sempurna sehingga pertumbuhannya melambat, bahkan tanaman dapat mati. Kerugian yang ditimbulkan oleh penyakit ini mencapai 70% [4].

Penting untuk mengidentifikasi penyebab penyakit lebih cepat agar dapat ditemukan solusi untuk melakukan penanganan. Mengidentifikasi bentuk dan warna bercak pada daun jagung adalah salah satu cara dapat dilakukan. Identifikasi penyebab masalah pada daun jagung masih dilakukan secara konvensional dengan cara melihat langsung pada daun jagung. Hal ini dapat menyebabkan kesalahan dalam mengidentifikasi penyakit, sebab gejala awal yang ditimbulkan beberapa penyakit memiliki kemiripan dalam segi warna maupun bentuk bercak pada daun jagung. Oleh karena itu harus diteliti lebih lanjut agar tidak melakukan kesalahan dalam penanganan. Terdapat beberapa peneliti dari berbagai bidang pengetahuan yang mencoba mengembangkan penelitian dan teknologi baru untuk membantu memecahkan masalah bagi para petani untuk dapat mengenali dan mengatasi penyakit tanaman dengan efektif. Salah satu teknologi populer untuk identifikasi penyakit tanaman adalah pengolahan citra. Pemrosesan citra komputer dapat diterapkan pada pembelajaran mesin untuk memungkinkan komputer memproses gambar dalam jumlah besar secara akurat.

Computer vision adalah bagian dari kecerdasan buatan yang melatih algoritma komputer untuk secara otomatis menganalisis, mengklasifikasikan, dan mendeteksi gambar baik dalam gambar video, maupun gambar *real-time* menggunakan jaringan syaraf dan pembelajaran mendalam [5]. *Deep learning* adalah cabang dari *machine learning* berbasis jaringan saraf tiruan. *Deep learning* ada karena data yang dibutuhkan untuk mengenal sebuah citra itu sangat banyak dan memiliki tingkat kompleksitas yang tinggi. Salah satu metode *deep learning* yang memiliki hasil paling signifikan untuk *image classification* adalah *Convolutional Neural Network* (CNN). Hal tersebut dikarenakan CNN berusaha meniru sistem pengenalan citra pada visual *cortex* manusia, sehingga memiliki kemampuan mengolah informasi citra [6].

Metode yang digunakan untuk penelitian ini adalah *AlexNet* yang merupakan arsitektur jaringan dari CNN. *AlexNet* adalah jaringan yang dibangun Karen Simonyan dan Andrew Zisserman [7]. Pada VGG terdapat 6 model dengan jumlah layer yang berbeda dengan memiliki 11 layer hingga 19 layer. Metode lain untuk meningkatkan akurasi dari CNN adalah menggunakan *transfer learning*. Transfer learning adalah arsitektur CNN yang sudah dilatih sebelumnya atau biasa disebut juga *pre-trained* model dengan suatu dataset sebelumnya yang nantinya bisa digunakan untuk melatih dataset lain.

Tahap segmentasi dilakukan untuk membagi citra menjadi 2 yaitu, objek dan *background*. Tahapan ini dilakukan untuk membangun citra yang bagus dikarenakan citra yang digunakan merupakan data primer local yang belum pernah digunakan sebelumnya. Maka dari itu, digunakan pengolahan citra melalui proses *HSV thresholding*. *Thresholding* merupakan proses mengubah citra menjadi citra biner, dimana citra hitam (*black*) bernilai 0 dan citra putih (*white*) bernilai 1, sehingga dapat memisahkan objek yang dibutuhkan dengan *background* berdasarkan nilai threshold yang ditentukan [8].

Hingga saat ini, sudah banyak dilakukan riset mengenai pengenalan jenis penyakit pada berdasarkan citra daun. Penelitian pertama menggunakan teknologi *deep learning* dalam mendeteksi dan mengklasifikasikan penyakit tanaman. Penelitian ini menggunakan data citra tanaman jagung dan padi, metode yang digunakan adalah *convolutional neural network AlexNet*. Tingkat akurasi yang didapatkan mencapai 91.83% [9].

Pada penelitian lainnya yang mengusulkan model klasifikasi menggunakan citra daun tomat yang dilatih menggunakan arsitektur CNN *AlexNet* dengan *transfer learning*. Proses segmentasi diterapkan melalui proses penghapusan latar belakang menjadi hitam menggunakan *hue*, *saturation*, dan *value* (*HSV*). Tingkat akurasi yang didapatkan mencapai 99,72% [10] . Hal ini terbukti model dengan segmentasi citra daun dengan *transfer learning* dapat dikembangkan untuk diterapkan pada klasifikasi dataset gambar daun lainnya.

Berdasarkan kedua contoh penelitian tersebut, dapat dilihat bahwa *Convolutional Neural Network* (CNN) dengan arsitektur *AlexNet* menunjukkan hasil performa yang baik dalam klasifikasi penyakit tanaman melalui citra daun. Oleh karena itu, penelitian ini menggunakan teknologi deep learning dengan arsitektur *AlexNet* dalam pengklasifikasian penyakit daun jagung ke dalam tiga kelas, yaitu daun sehat (*healthy*), bercak daun (*leaf blight*) dan karat daun (*common rust*) dengan menggunakan tahapan *preprocessing* data melalui proses *resize*, *segmentasi*, *augmentasi* dan *k-fold cross validation*. Penelitian ini menggunakan teknik *transfer learning* arsitektur *AlexNet* yang telah dimodifikasi. Alasan peneliti memilih *AlexNet* yang telah dimodifikasi karena pada penelitian sebelumnya, *AlexNet* mampu menghasilkan performa akurasi yang bagus [11].

1.2 Perumusan Masalah

1.2.1 Permasalahan

Dari permasalahan, jenis penyakit pada daun jagung ini terkadang tidak teridentifikasi oleh orang yang awam, sehingga akan berakibat kesalahan dalam mengenali jenis penyakit dan penanganannya. Oleh karena itu, penelitian ini melakukan klasifikasi citra penyakit tanaman jagung yang teridentifikasi pada bagian daun. Maka dibutuhkan metode untuk membentuk sistem kecerdasan buatan agar mempermudah klasifikasi.

1.2.2 Metode Penelitian

Metode yang digunakan ialah *Convolutional Neural Network* (CNN) arsitektur *AlexNet* yang telah dimodifikasi dalam mendapatkan hasil klasifikasi penyakit tanaman jagung berdasarkan citra daun.

1.2.3 Pertanyaan Penelitian

1. Berapakah performa (*accuracy*, *sensitivity*, *specificity*) yang didapatkan dari proses *training* dan *testing* model klasifikasi penyakit daun jagung dengan menggunakan modifikasi *AlexNet* dengan *transfer learning*?
2. Bagaimana pengaruh *learning rate* yang didapatkan dari proses *training* dan *testing* model

klasifikasi penyakit daun jagung dengan menggunakan *AlexNet* dan modifikasi *AlexNet* dengan *transfer learning*?

1.3 Tujuan dan Manfaat

1.3.1 Tujuan

1. Mengetahui performa (*accuracy*, *sensitivity*, *specificity*) yang didapatkan dari proses *training* dan *testing* model klasifikasi penyakit jagung dengan menggunakan modifikasi *AlexNet* dengan *transfer learning*.
2. Mengetahui pengaruh *learning rate* yang didapatkan dari proses *training* dan *testing* model klasifikasi penyakit jagung dengan menggunakan modifikasi *AlexNet* dengan *transfer learning*.

2.3.1 Manfaat

Penelitian ini memiliki manfaat untuk memudahkan suatu pihak dalam mengembangkan program yang mampu mendeteksi dan mengklasifikasikan penyakit daun jagung. Diharapkan hasil dari penelitian ini dapat membantu petani dalam meningkatkan produksi jagung dan meminimalkan kerugian akibat serangan penyakit pada tanaman jagung.

1.4 Batasan Masalah

Adapun batasan permasalahan penelitian ini yaitu:

1. Data yang digunakan adalah dataset daun jagung yang diambil secara langsung di daerah persawahan Desa Socah dan Desa Benangkah, Bangkalan.
2. Dataset yang dipakai berformat .jpg dan berukuran 224x224 piksel setelah dilakukan *resize* citra. Gambar diambil menggunakan kamera handphone yang memiliki resolusi kamera 64 Megapixel.
3. Total data yang digunakan pada masing-masing kelas berjumlah 95 dengan total data citra asli 285. Untuk memperbanyak jumlah citra dilakukan augmentasi pada data *training* yang berjumlah 228 dan didapatkan 1368 data *training*. Data *testing* yang digunakan 57 data.
4. Teknik augmentasi yang digunakan diantaranya *rotation*, *zoom_range*, *shear_range*, *horizontal flip*, dan *width_shift_range*.
5. Output yang dihasilkan berupa klasifikasi pada citra penyakit daun jagung dengan kelas daun sehat (*healthy*), bercak daun (*leaf blight*), karat daun (*common rust*).
6. Hyperparameter yang digunakan adalah :

- a. *Epoch* = 30
- b. *Batch size* = 32
- c. *Optimizer* = Adam
- d. *Learning rate* = 0.1, 0.01, 0.001

1.5 Metodologi Penulisan Skripsi

Metodologi penulisan skripsi disusun mengikuti aturan penulisan skripsi Sistem Informasi Universitas Trunojoyo Madura. Berikut merupakan metodologi penulisan skripsi pada penelitian ini.

BAB I PENDAHULUAN

Metodologi penulisan skripsi disusun mengikuti aturan penulisan skripsi Sistem Informasi Universitas Trunojoyo Madura. Berikut merupakan metodologi penulisan skripsi pada penelitian ini.

BAB II TINJAUAN PUSTAKA

Metodologi penulisan skripsi disusun mengikuti aturan penulisan skripsi Sistem Informasi Universitas Trunojoyo Madura. Berikut merupakan metodologi penulisan skripsi pada penelitian ini.

BAB III PERANCANGAN SISTEM

Pada bab ini membahas perancangan sistem yang mencakup jenis penelitian, tahapan penelitian, analisis kebutuhan sistem, skenario pengujian, jadwal penelitian.

BAB IV HASIL DAN PEMBAHASAN

Menjelaskan tentang hasil yang telah didapatkan dari proses uji coba, seperti hasil uji coba training dan testing klasifikasi penyakit daun jagung yang dilakukan berdasarkan skenario uji coba.

BAB V PENUTUP

Menjelaskan tentang kesimpulan dari hasil penelitian yang sesuai dengan pertanyaan penelitian dan saran metode yang diusulkan.

DAFTAR PUSTAKA

LAMPIRAN

BAB II

TINJAUAN PUSTAKA

2.1 Penyakit Tanaman padi

Pada umumnya penyakit yang biasa dialami pada tanaman jagung dapat menyerang bagian daun, akar, tongkol hingga batang. Hal ini disebabkan oleh jamur, bakteri, virus, mikroplasma, nematoda, dan parasit. Hal tersebut biasanya disebabkan oleh angin dan air yang ditentukan oleh cuaca. Pada penelitian ini, dataset yang digunakan adalah 2 penyakit pada daun jagung. Diantaranya bercak daun (*leaf blight*) dan karat daun (*common rust*). Berikut penjelasan penyakit daun jagung.

2.1.1 Blast

Penyakit *blast* adalah penyakit yang menyebabkan daun padi memiliki bercak berbentuk lonjong dan berbentuk belah ketupat pada daun padi yang terlihat pada Gambar 2.1.



Gambar 2.1 Penyakit *Blast* daun

2.1.2 Penyakit Bercak Daun (*Leaf Blight*)

Penyakit ini disebabkan oleh cendawan *Helminthosporium turcicum* dengan gejala tampaknya bercak memanjang dan teratur berwarna kuning dan dikelilingi warna coklat. Daun kemudian akan berbuah menjadi coklat kekuning-kuningan, hingga coklat tua. Gejala awal pada bercak daun mungkin sulit diidentifikasi sebagai bercak daun abu-abu karena menyerupai mata dan karat pada umumnya dapat dilihat pada Gambar 2.2.



Gambar 2. 2 Penyakit Bercak Daun

2.1.3 Penyakit Tungro

Penyakit ini ditularkan melalui gigitan wereng, terutama wereng hijau dan wereng coklat. Gejala tungro mencakup daun padi yang menguning, pembesaran pembuluh kayu pada batang, dan padi yang tumbuh lebih pendek dari biasanya dapat dilihat pada Gambar 2.3



Gambar 2.3 Penyakit Tungro

2.2 Pengolahan Citra Digital

Pengolah citra digital merupakan teknik atau ilmu yang mempelajari beberapa cara untuk mengolah gambar atau citra. Adapun jenis citra yang terbagi menjadi 3, diantaranya:

2.2.1 Citra Berwarna

Dalam konteks citra warna, menggambarkan proses pembentukan citra berwarna melalui kombinasi tiga warna dasar: *red*, *green*, dan *blue* (RGB). Setiap piksel dalam citra direpresentasikan dengan 24 bit warna, di mana masing-masing warna dasar memiliki nilai rentang dari 0 hingga 255. Nilai ini mencerminkan intensitas cahaya untuk masing-masing warna, dan kombinasi nilai-nilai ini menghasilkan spektrum warna yang beragam pada setiap piksel, membentuk citra warna yang dilihat secara visual.

2.2.2 Citra Grayscale

Citra *grayscale* (skala keabuan) adalah citra yang setiap pikselnya mewakili skala keabuan. Pada citra *grayscale*, setiap piksel mewakili derajat keabuan dengan nilai 0 mewakili warna hitam (*black*) dan 255 mewakili warna putih (*white*) [11]. Dalam rentang nilai 0 sampai 255, bahwa setiap piksel memiliki ukuran 8bit atau 1bit. Citra grayscale dinyatakan dalam fungsi (2.1) berikut :

$$f(x, y) = \{0, \dots, 255\} \quad (2.1)$$

Pada gambar 2.4 di bawah ini menunjukkan warna skala keabuan pada citra 8 bit, yaitu dari warna hitam, keabuan dan putih pada setiap nilai bagian merah, hijau, dan biru.



Gambar 2.4 Palet *grayscale* pada nilai bagian *red*, *green* dan *blue*

2.2.3 Citra Binary Image

Citra biner merupakan bagian dari citra *grayscale* yang hanya memiliki dua level keabuan, yaitu 0 mewakili warna hitam dan 1 mewakili warna putih, sehingga setiap piksel dari citra biner dikodekan dengan hanya menggunakan 1 bit. Citra biner dihitung dengan menggunakan nilai ambang (*threshold*). Bila nilai piksel lebih kecil daripada batas ambang maka nilai tersebut diubah menjadi 0 (*black*), dan bila lebih besar atau sama dengan nilai ambang maka nilai tersebut diubah menjadi 1 (*white*) [12].

$$I_{Bin}(x, y) = \begin{cases} 0 & I_{BW}(x, y) < T \\ 255 & I_{BW}(x, y) \geq T \end{cases} \quad (2.2)$$

Bentuk *floating point*

$$I_{Bin}(x, y) = \begin{cases} 0 & I_{BW}(x, y) < T \\ 1 & I_{BW}(x, y) \geq T \end{cases} \quad (2.3)$$

Keterangan :

T = adalah nilai dari *threshold*

$I_{BW}(x, y)$ = adalah nilai dari pixel gray titik (x, y)

$I_{Bin}(x, y)$ = adalah nilai dari pixel binary titik (x, y)

2.3 Operasi Morfologi

Operasi morfologi dalam pemrosesan citra adalah serangkaian teknik yang melibatkan manipulasi bentuk dan struktur objek dalam citra. Dengan menggunakan elemen dasar, seperti dilasi, erosi, opening, dan closing, operasi morfologi dapat meningkatkan atau mengurangi ukuran objek, menghilangkan noise, dan memperbaiki kontur objek. Teknik ini digunakan secara luas dalam analisis citra, segmentasi objek, dan ekstraksi fitur[13].

Dalam penelitian ini, penulis menggunakan operasi closing morfologi. Teknik ini menggabungkan operasi erosi dan dilatasi yang dilakukan secara bergantian untuk mengisi celah pada bagian yang ditangkap. Operasi dilasi dilakukan untuk mengubah piksel-piksel pada gambar yang menjadi bagian latar belakang dan menjadikannya bagian dari objek itu sendiri berdasarkan elemen B. Sedangkan operasi dilatasi dilakukan untuk memperkecil ruang pada objek, memindahkan sebagian objek ke latar belakang berdasarkan elemen inti B.

2.4 Deep Learning

Deep learning merupakan cabang dari *machine learning* yang memiliki jaringan tersendiri, yaitu jaringan saraf tiruan yang dapat belajar dan beradaptasi terhadap sejumlah data besar [14]. Pada dasarnya, *deep learning* adalah jaringan saraf dengan tiga atau lebih lapisan. Deep learning bekerja dengan *artificial neural networks* yang dibentuk untuk meniru cara berpikir manusia. Sistem akan belajar sendiri dari sejumlah besar data yang disediakan.

Keberhasilan yang sangat bagus dari penelitian *deep neural network*, teknik *deep learning* untuk mendeteksi, mengklasifikasi, dan mengidentifikasi objek dalam gambar. Jaringan saraf konvolusional merupakan salah satu metode terpopuler dalam kategori jaringan saraf dalam, yang telah terbukti sangat efektif di berbagai bidang seperti pengenalan dan klasifikasi gambar.

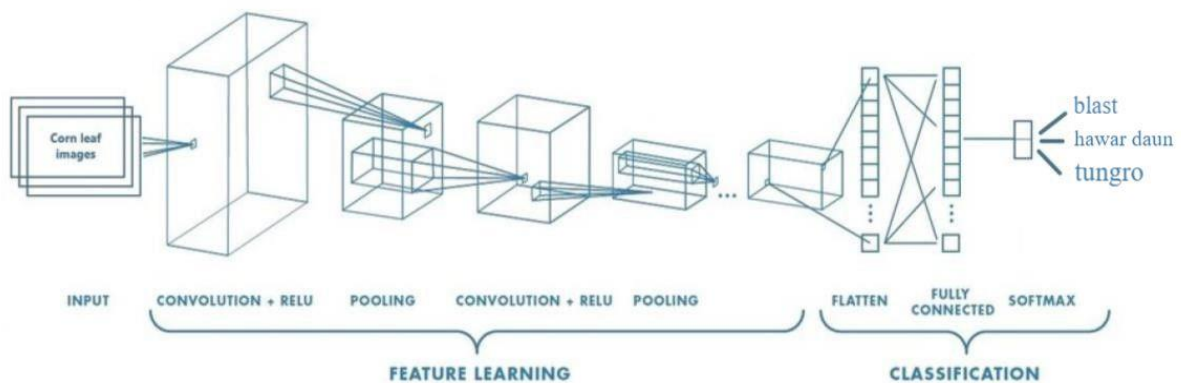
Saat ini, teknologi *deep learning* mendorong pengembangan kecerdasan buatan. *Deep learning* banyak diterapkan di banyak bidang,. Contoh populer dari implementasi *deep learning* yaitu *self-driving car*, *google translate*, dan deteksi wajah dari fitur *Google Photo*.

2.5 Transfer Learning

Transfer learning merupakan metode yang bekerja dengan memanfaatkan arsitektur jaringan saraf yang telah ada. Transfer learning memanfaatkan model yang sudah di *training* sebelumnya (*pre-trained model*) yang digunakan untuk mengklasifikasikan dataset yang baru sehingga tidak perlu melakukan training data dari awal [15].

2.6 Convolutional Neural Network

Convolutional Neural Network (CNN) merupakan salah satu algoritma *deep learning* yang dapat menerima input gambar dan sebagai sumber mengenali gambar yang membedakan antara setiap gambar yang ada. Algoritma CNN mendapatkan hasil yang paling signifikan dalam pengenalan citra digital dikarenakan implementasi CNN berdasarkan pengenalan citra pada *visual cortex* manusia. Secara teknis, CNN adalah sebuah arsitektur dapat dilatih dan terdiri dari beberapa tahapan. Masukan (*input*) dan keluaran (*output*) dari setiap tahapan terdiri dari beberapa *array* yang disebut *feature map* [16]. Gambar 2.6 menunjukkan arsitektur CNN.



Gambar 2.5 Arsitektur *Convolutional Neural Network*

Berdasarkan pada gambar ada tahapan awal pada arsitektur CNN yaitu, tahapan konvolusi pada *convolutional layer* melakukan konvolusi output dari layer sebelumnya. Konvolusi melakukan perhitungan dengan sebuah kernel memiliki ukuran tertentu. Kemudian dilanjutkan dengan fungsi aktivasi yaitu, *ReLU* (*Rectifier Linear unit*) lalu setelah itu masuk pada tahapan pooling pada pooling layer. Tahapan tersebut diulang hingga mendapatkan feature map yang cukup untuk dilanjutkan pada tahapan *fully-connected layer*[17].

2.6.1 Convolutional Layer

Proses *convolutional layer* digunakan untuk melakukan ekstraksi ciri dari tiap citra. Proses ini memiliki beberapa komponen antara lain, filter, *stride* dan *zero padding*. Filter merupakan matriks yang telah dibagi menjadi ukuran lebih kecil yaitu $n \times n$, pada umumnya ukuran filter bisa 3×3 , 5×5 , atau 7×7 . Filter berfungsi untuk mengenali objek. Pada filter terdapat operasi dot yang merupakan perkalian antara bobot dan matriks hasil dari setiap piksel citra [18]. Piksel yang dikalikan akan dijumlahkan menjadi fitur. Rumus konvolusi dapat dilihat pada persamaan (2.4)

$$f_i(x,y) = \sum k * f_{i-1}(x,y) \quad (2.4)$$

Keterangan :

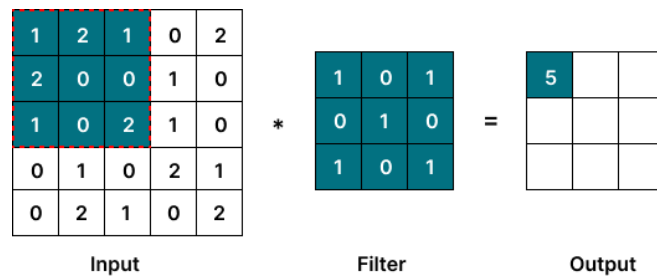
$f_i(x,y)$ = hasil konvolusi (*feature map*).

k = kernel.

$f_{i-1}(x,y)$ = hasil konvolusi (*feature map*) sebelumnya.

(x,y) = posisi piksel pada matriks input gambar.

Contoh proses konvolusi pada *convolutional layer* dengan kernel 3x3 tanpa padding yang ditunjukkan pada Gambar 2.6.



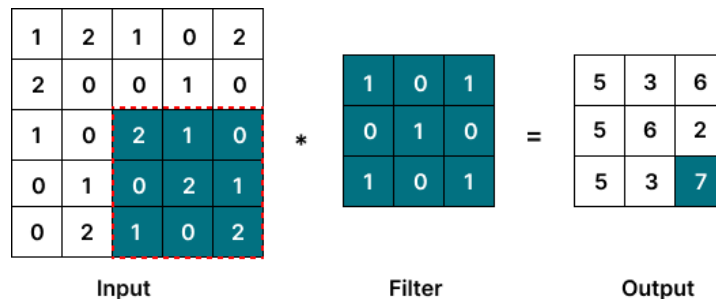
Gambar 2.6 Proses konvolusi layer

Nilai yang dihasilkan oleh baris ke 1 dan kolom ke 1 pada *feature map* di dapat dari:

$$(1 \times 1) + (2 \times 0) + (1 \times 1) + (2 \times 0) + (0 \times 1) + (0 \times 0) + (1 \times 1) + (0 \times 0) + (2 \times 1)$$

$$= 1 + 1 + 1 + 2 = 5$$

Setelah menyelesaikan proses konvolusi pada baris ke 1 dan kolom ke 1, proses selanjutnya akan dilakukan pergeseran atau yang bisa disebut *stride* dengan nilai 1. Proses konvolusi tersebut akan terus bergeser hingga semua bagian pada *feature map* terkonvolusi. Hasil konvolusi selanjutnya ditunjukkan pada Gambar 2.7.



Gambar 2.7 Proses konvolusi kolom terakhir

Saat dilakukan proses konvolusi, tentunya ukuran citra hasil konvolusi menjadi lebih kecil daripada ukuran citra input. Semakin besar nilai *stride* digunakan maka semakin kecil juga

ukuran citra konvolusinya. Kecilnya ukuran citra hasil konvolusi, dapat berdampak pada informasi yang didapatkan. Oleh karena itu, penambahan *zero padding* dapat dilakukan untuk mencegah terbuangnya informasi. *Padding* atau yang biasa disebut *zero padding* merupakan parameter yang menentukan jumlah piksel berisi nilai 0 pada setiap sisi luar citra input untuk memanipulasi dimensi output pada setiap proses konvolusi. Sehingga bisa menggunakan konvolusi yang lebih dalam sehingga lebih banyak *features* yang berhasil diekstrak. *Zero padding* berukuran 1x1 pada matrix 5x5 seperti yang ditunjukkan pada Gambar 2.8.

0	0	0	0	0	0	0
0	1	2	1	0	2	0
0	2	0	0	1	0	0
0	1	0	0	1	0	0
0	0	1	0	2	1	0
0	0	2	1	0	2	0
0	0	0	0	0	0	0

Gambar 2.8 Matriks dengan *zero padding*

Setiap proses konvolusi tentunya akan menghasilkan citra output proses konvolusi. Ukuran output hasil konvolusi ditentukan oleh tiga parameter yaitu dimensi *filter*, *stride* dan *padding*. Berikut merupakan persamaan dari output hasil konvolusi pada persamaan (2.5)

$$Output = \frac{W+N 2P}{S} + 1 \quad (2.5)$$

Keterangan :

W = Panjang/Tinggi input

N = Panjang/Tinggi filter

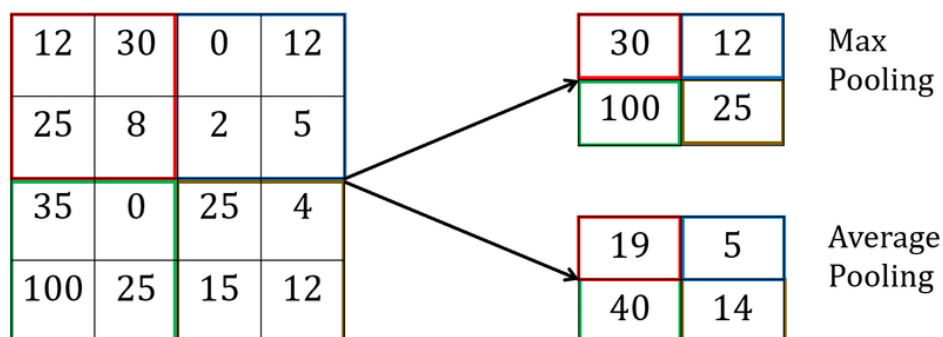
P = Zero padding

S = Stride

2.6.2 Pooling Layer

Pooling layer merupakan proses setelah lapisan konvolusi. Fungsi dari *pooling layer* adalah mengurangi dimensi pada peta fitur dengan operasi *downsampling* pada shape suatu citra. Tujuannya adalah untuk mempertahankan informasi penting dari hasil output, sehingga mempercepat proses komputasi. Ada dua macam *pooling layer* yang sering digunakan, yaitu *average pooling* dan *max pooling*. Pada penelitian ini penulis menggunakan *max pooling* dan *global average pooling*.

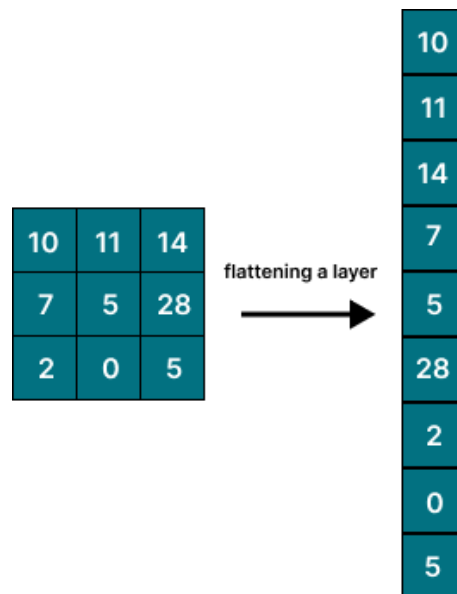
Average pooling dilakukan dengan cara membagi *output* lapisan konvolusi menjadi beberapa grid kecil, kemudian mengambil rata-rata dari setiap grid untuk membuat matriks gambar baru yang berukuran lebih kecil. *Max pooling* dihitung dengan cara membagi output lapisan konvolusi menjadi beberapa kisi kecil, dan kemudian mengambil nilai maksimum atau nilai yang paling besar dari tiap-tiap grid untuk membuat matriks citra baru yang berukuran kecil, seperti yang diperlihatkan oleh Gambar 2.10. *Grid* yang berwarna merah, biru, kuning dan hijau adalah kelompok kisi yang nilainya diambil nilai maksimumnya. Gambar 2.9 menunjukkan contoh dari *pooling layer*.



Gambar 2.9 Pooling layer

2.6.3 Flatten

Flatten adalah operasi dalam jaringan saraf tiruan, khususnya dalam konteks *Convolutional Neural Networks* (CNN), yang mengubah data multidimensi (seperti matriks atau tensor) menjadi vektor satu dimensi. Tujuannya adalah untuk mempersiapkan data agar dapat diproses oleh lapisan *fully connected* (juga dikenal sebagai *dense layer*) dalam jaringan saraf. Ilustrasi pembentukan ulang *feature map* ditunjukkan pada Gambar 2.10.

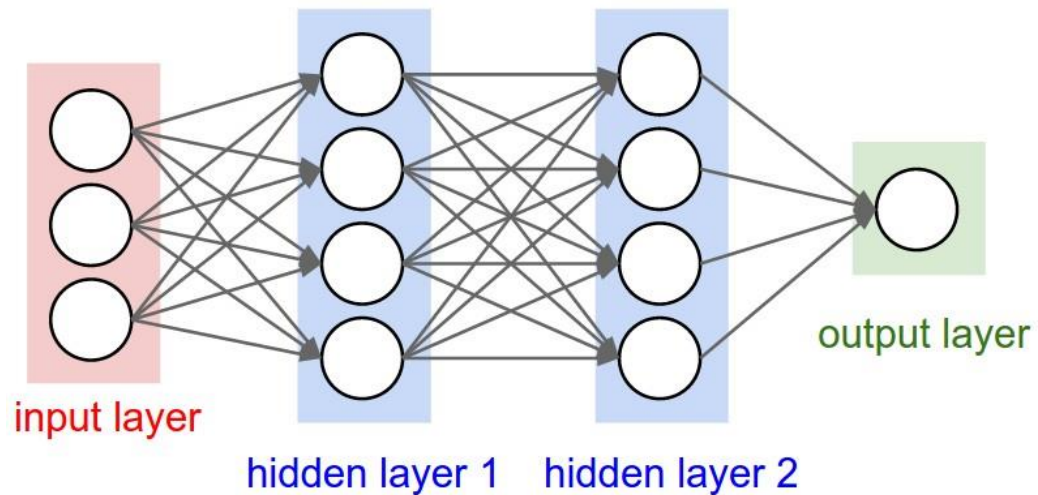


Gambar 2.10 *Flatten*

2.6.4 Fully Connected Layer

Umumnya, *fully connected layer* terletak di akhir CNN arsitektur. Lapisan ini digunakan untuk mengklasifikasikan gambar ke dalam label dengan hasil dari lapisan sebelumnya. Nilai-nilai yang dihasilkan dari konvolusi dan penggabungan diratakan menjadi satu vektor nilai. Gambar 2.11 menunjukkan alur kerja *fully connected layer* ketika nilai input masuk ke lapisan awal perceptron. Nilai-nilai tersebut dikalikan dengan bobot dan dilewatkan melalui fungsi aktivasi, kemudian diteruskan ke lapisan output, di mana setiap *perceptron* mewakili label yang diklasifikasikan.

Fungsi *dense* ditambahkan untuk menambahkan layer yang *fully connected*. Parameter pertama menggambarkan jumlah node yang harus ada di lapisan tersembunyi, nilainya antara jumlah *input* node dan *output* node. Parameter kedua *layer output* mewakili fungsi aktivasi yang digunakan dengan fungsi *softmax*.



Gambar 2.11 *Fully connected layer*

Pada contoh Gambar 2.11 fungsi *input layer* merupakan dari *pooling layer* atau *activation layer* sebelumnya, sedangkan *output layer* yang akan menghasilkan dari probabilitas dari setiap kelas.

2.6.5 Rectified Linear Unit (ReLU)

ReLU merupakan fungsi aktivasi yang paling banyak digunakan pada lapisan konvolusi. ReLU digunakan untuk menghilangkan nilai negatif pada *feature map* [18]. Seperti yang ditunjukkan pada persamaan (2.6)

$$f(x) = \max(0, x) \quad (2.6)$$

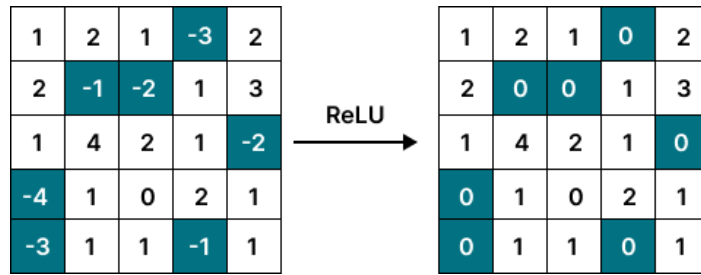
Keterangan :

$f(x)$ = Fungsi dari aktivasi ReLU

x = Hasil dari proses pooling

$\max$ = Fungsi Maximum

Fungsi ReLU mengambil nilai piksel pada citra input kemudian melakukan aktivasi dan mengembalikannya. Layer ReLU mengubah nilai piksel x menjadi nilai 0 jika nilai x tersebut negatif dan tetap mempertahankan nilai x jika nilai x tersebut positif. Gambar 2.13 Memperlihatkan contoh proses aktivasi ReLU.



Gambar 2.12 Fungsi aktivasi ReLU

2.6.6 Softmax

Softmax adalah fungsi aktivasi yang digunakan dalam konteks jaringan saraf tiruan, khususnya di lapisan output dari model klasifikasi. Fungsi ini mengubah sekumpulan nilai menjadi distribusi probabilitas, sehingga setiap nilai dalam output mencerminkan probabilitas relatif dari setiap kelas. Fungsi aktivasi softmax dapat dilihat pada persamaan (2.7)

$$F_z(Z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (2.7)$$

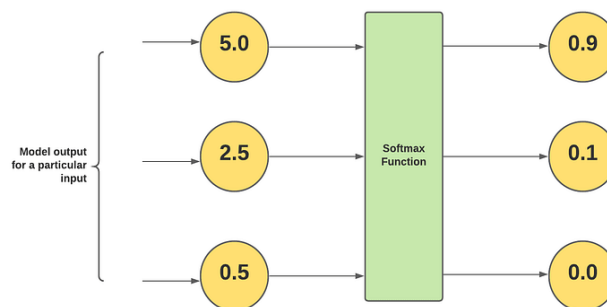
Keterangan :

$F_z(Z)$ = Probabilitas kelas yang dihasilkan oleh fungsi softmax.

e^{z_j} = Nilai eksponensial dari input pada posisi ke j.

$\sum_k e^{z_k}$ = Jumlah dari eksponensial semua elemen input untuk normalisasi probabilitas.

Pada contoh Gambar 2.14 memiliki model yang telah dilatih untuk tugas klasifikasi yang melibatkan tiga kelas berbeda. Setelah melakukan komputasi pada data masukan tertentu, model memberikan tiga nilai keluaran, yaitu 5.0, 2.5, dan 0.5. Kemudian mengonversi nilai-nilai ini menjadi probabilitas yang menunjukkan sejauh mana setiap kelas mungkin menjadi kelas yang benar. Hasil dari langkah ini adalah probabilitas untuk masing-masing kelas.



Gambar 2.13 Fungsi aktivasi *softmax*

2.7 Asitektur AlexNet

AlexNet adalah arsitektur *Convolutional Neural Network* (CNN) yang diperkenalkan oleh Alex Krizhevsky, Ilya Sutskever, dan Geoffrey Hinton melalui makalah "*ImageNet Classification with Deep Convolutional Neural Networks*" pada tahun 2012. Arsitektur ini berhasil memenangkan kompetisi *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) 2012 dengan selisih yang signifikan, sehingga menjadi tonggak penting dalam perkembangan deep learning. Berikut adalah penjelasan formal mengenai arsitektur *AlexNet*:

1. Input Layer:

- Ukuran input: 224 x 224 x 3 (panjang, lebar, dan jumlah saluran warna RGB).
- Normalisasi: Nilai piksel gambar dinormalisasi untuk meningkatkan stabilitas dan kinerja pelatihan.

2. Layer 1 (Convolutional Layer + ReLU + Max-Pooling):

- Convolutional Layer:
 - Filter: 96 filter ukuran 11 x 11 dengan stride 4.
 - Padding: Tidak ada padding (valid padding).
 - Activation Function: Rectified Linear Unit (ReLU) digunakan untuk menambahkan non-linearitas.
- Max-Pooling:
 - Ukuran kernel: 3 x 3.
 - Stride: 2.
 - Output: 55 x 55 x 96.

3. Layer 2 (Convolutional Layer + ReLU + Max-Pooling):

- Convolutional Layer:
 - Filter: 256 filter ukuran 5 x 5 dengan stride 1.

- Padding: 2 (same padding).
- Activation Function: ReLU.
- Max-Pooling:
- Ukuran kernel: 3 x 3.
- Stride: 2.
- Output: 27 x 27 x 256.

4. Layer 3 (Convolutional Layer + ReLU):

- Convolutional Layer:
- Filter: 384 filter ukuran 3 x 3 dengan stride 1.
- Padding: 1 (same padding).
- Activation Function: ReLU.
- Output: 13 x 13 x 384.

5. Layer 4 (Convolutional Layer + ReLU):

- Convolutional Layer:
- Filter: 384 filter ukuran 3 x 3 dengan stride 1.
- Padding: 1 (same padding).
- Activation Function: ReLU.
- Output: 13 x 13 x 384.

6. Layer 5 (Convolutional Layer + ReLU + Max-Pooling):

- Convolutional Layer:
- Filter: 256 filter ukuran 3 x 3 dengan stride 1.
- Padding: 1 (same padding).
- Activation Function: ReLU.

- Max-Pooling:
- Ukuran kernel: 3 x 3.
- Stride: 2.
- Output: 6 x 6 x 256.

7. Layer 6 (Fully Connected Layer + ReLU + Dropout):

- Fully Connected Layer: 4096 neurons.
- Activation Function: ReLU.
- Dropout: Probabilitas dropout 0.5 untuk mengurangi overfitting.

8. Layer 7 (Fully Connected Layer + ReLU + Dropout):

- Fully Connected Layer: 4096 neurons.
- Activation Function: ReLU.
- Dropout: Probabilitas dropout 0.5.

9. Layer 8 (Fully Connected Layer + Softmax):

- Fully Connected Layer: 1000 neurons (untuk 1000 kelas dalam dataset ImageNet).
- Activation Function: Softmax untuk menghasilkan probabilitas klasifikasi.

Fitur Utama *AlexNet*:

- ReLU Activation: Penggunaan fungsi aktivasi ReLU di setiap *convolutional layer*, yang mempercepat konvergensi pelatihan dibandingkan dengan fungsi aktivasi tanh atau sigmoid.
- Dropout: Teknik regularisasi untuk mengurangi overfitting dengan secara acak menonaktifkan neuron selama pelatihan.
- Data Augmentation: Teknik augmentasi data seperti cropping acak, flipping, dan perubahan intensitas untuk meningkatkan ukuran dan variasi dataset.
- GPU Utilization: Pelatihan *AlexNet* dilakukan dengan dua GPU, membagi jaringan menjadi dua bagian untuk paralelisasi komputasi.

Arsitektur *AlexNet* memperkenalkan beberapa inovasi yang menjadi standar dalam pengembangan CNN selanjutnya dan telah terbukti sangat efektif dalam berbagai tugas pengenalan gambar.

2.8 Augmentasi Citra

Augmentasi citra (*image augmentation*) dalam konteks *Convolutional Neural Networks* (CNN) adalah teknik untuk meningkatkan variasi dataset gambar tanpa benar-benar mengumpulkan gambar baru. Ini dilakukan dengan menerapkan berbagai transformasi pada gambar asli, seperti rotasi, translasi, penskalaan, dan perubahan warna. Tujuan utamanya adalah untuk meningkatkan kinerja model dan generalisasi dengan menciptakan lebih banyak variasi dari data pelatihan yang ada. Proses augmentasi terdiri dari beberapa tahapan yaitu, *flip*, *random crop*, *random rotation*, *random resize crop*, dan *shear*.

Tabel 2.2 Macam-macam teknik augmentasi data

Teknik	Keterangan
<i>Flip</i>	Digunakan untuk membalik posisi suatu citra secara horizontal maupun vertikal
<i>Random Rotation</i>	Digunakan untuk merotasi sebuah citra
<i>Shear</i>	Digunakan untuk memiringkan posisi citra
<i>Zoom</i>	Digunakan memperbesar ukuran citra

2.9 Evaluasi Kinerja Model

2.9.1 K-fold Cross Validation

Pada penelitian ini metode yang digunakan pada saat tahap *training* ialah *k-fold cross validation* dan menghitung akurasi dengan menggunakan *confussion matrix*. Cara kerja dari *k-fold cross validation* yaitu dengan membagi sampel data secara acak dan mengelompokkan data tersebut sebanyak nilai K *k-fold* [21]. Kemudian salah satu kelompok *k-fold* tersebut dibagi menjadi data *training* dan data *validation* untuk mengevaluasi hasil model yang dibuat pada proses *training* sebelumnya. Pada penelitian ini menggunakan *k-fold validation* $k=5$, di mana data yang digunakan adalah 10 bagian sebagai *training* dan 1 *validation*.

2.9.2 Matrix Evaluasi

Dalam tahap evaluasi *confusion matrix* digunakan sebagai solusi untuk menghitung prediksi nilai akurasi yang dihasilkan dari proses *training* dan *testing*. Terdapat empat istilah yang

digunakan untuk merepresentasikan hasil klasifikasi yaitu, *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Contoh *multiclass confusion matrix* ditunjukkan pada Gambar 2.15 berikut [22].

		Predicted Classification		
Actual Classification	Classes	A	B	C
	A	TN (True Negative)	FP (False Positive)	TN (True Negative)
	B	FN (False Negative)	TP (True Positive)	FN (False Negative)
	C	TN (True Negative)	FP (False Positive)	TN (True Negative)

Gambar 2.15 *Multiclass confusion matrix*

Keterangan :

True Positive (TP) : Jumlah hasil klasifikasi benar dari kelas aktual positif.

True Negative (TN) : Jumlah hasil klasifikasi benar dari kelas aktual negatif.

False Positive (FP) : Jumlah hasil klasifikasi salah dari kelas aktual positif.

False Negative (FN) : Jumlah hasil klasifikasi salah dari kelas aktual negatif.

Pada Gambar menunjukkan contoh perhitungan yang mengikuti aturan *multiclass confusion matrix*, dimana kelas target dan kelas prediksi berjumlah 3 sesuai dengan penelitian yang akan dilakukan. Dengan adanya confusion matrix akan ditemukan nilai *accuracy*, *sensitivity*, dan *specificity* seperti persamaan berikut ini :

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \times 100 \quad (2.8)$$

Persamaan (2.8) merupakan perhitungan rata-rata nilai *accuracy* untuk menunjukkan tingkat efektivitas perkelas dari sebuah klasifikasi.

$$Sensitivity = \frac{TP}{TP+FN} \times 100 \quad (2.9)$$

Persamaan (2.9) merupakan perhitungan rata-rata nilai *sensitivity* berguna untuk mengetahui ketepatan sistem untuk melihat berapa persen data dari salah satu label yang diprediksi benar.

$$Specificity = \frac{TN}{TN+FP} \times 100 \quad (2.10)$$

Persamaan (2.10) merupakan perhitungan rata-rata nilai *specificity* untuk perbandingan jumlah label yang diprediksi sebagai negatif dengan pada label yang memang negatif Persamaan (3.2) merupakan perhitungan rata-rata nilai *specificity* untuk perbandingan jumlah label yang diprediksi sebagai negatif dengan pada label yang memang negatif

2.10 Penelitian Terkait

Sebelumnya terdapat beberapa penelitian mengenai pengklasifikasian penyakit tanaman padi. Penelitian terkait ini diharapkan dapat digunakan sebagai pedoman pada penelitian ini.

Penelitian sebelumnya adalah Klasifikasi Penyakit Daun padi Menggunakan Metode Convolutional Neural dengan arsitektur Network AlexNet, pada penelitian ini sejumlah 4198 dataset citra daun padi yang digunakan meliputi Gray leaf spot, common rust, Blight dan Healthy dengan penelitian menggunakan AlexNet sebagai arsitektur dan CNN sebagai metode dapat memberikan akurasi yang maksimal sebesar 90% sehingga penulis memberikan kesimpulan metode yang dipilih dapat mengklasifikasikan beberapa jenis penyakit daun pada tanaman padi secara baik[21]

Penelitian kedua yaitu, Pengaruh Arsitektur Convolutional Neural Network Untuk Klasifikasi Penyakit daun Tomat dataset yang digunakan telah diklasifikasi terlebih dahulu menjadi 3 kelompok yaitu bacterial spot, early blight and leaf mold and preprocessing data dilakukan dengan mengompres citra pada dataset yang semula 256x256 px menjadi 224x224 px setelah dilakukan pengujian, hasil menunjukkan model CNN konvensional perlu digabungkan dengan arsitektur yang sudah diuji untuk dapat menghasilkan akurasi yang maksimal[22].

Penelitian selanjutnya yang dibuat oleh Monorawul Islam dkk pada tahun 2023 membandingkan arsitektur VGG16, ALEXNET, Xception dan Convolutional neural Networks mampu mengklasifikasi penyakit daun kapas. Setelah melakukan pengujian arsitektur Convolutional neural Networks memiliki performa yang baik sampai memperoleh nilai akurasi sebesar 97,83 % dan nilai presisi 97,88%[23].

Author	Dataset	Metode	Kelas	Performa
Qudsiah Nur Azizah, 2023	4198 citra daun jagung yang mencakup 4 kelas yaitu Blight, Healthy, Gray Leaf Spot, dan Common Rust	<i>Convolutional Neural Network (CNN) dengan arsitektur AlexNet</i>	3 (Blast, tungro, dan hawar daun)	90%
Rizky Arya Kurniawan, Andi Sunyoto, dan Asro Nasiri,	Dataset memiliki 4079 gambar penyakit daun tomat terdiri dari 3 kelas penyakit diantaranya bacterial spot, early blight dan leaf mold.	<i>Convolutional neural Networks, Arsitektur yang digunakan adalah InceptionV3, ResNet50 dan MobileNet.</i>	bacterial spot, early blight dan leaf mold.	99,67%
Monowarul Islam, Alamin Talukder, Rahul Amain S, Ashraf Uddin, Selina Sharmin, Selim Al Mamun, dan Sumon Kumar. 2023	2310 data total gambar daun kapas	VGG16, ALEXNET, <i>Convolutional neural Networks</i> , dan <i>Xception</i>	4 (Fresh leaf, fresh tree, disease leaf, disease tree)	97,83%