BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Sebelumnya

Dalam kaitannya dengan penelitian yang dilakukan oleh penulis, diperlukan adanya referensi dasar dari penelitian sebelumnya. Dengan adanya referensi dari penelitian sebelumnya dapat membantu penulis untuk melakukan penilitian yang berkaitan dan juga menghindari adanya duplikasi penelitian. Sehingga penelitian yang dilakukan oleh penulis dapat berkontribusi dalam bidang ilmu pengetahuan.

Beberapa penelitian yang telah dilakukan mengenai klasifikasi penyakit daun jagung menggunakan algoritma *Convolutional Neural Network* (CNN) seperti penelitian yang berjudul *Corn Leaf Disease Clasification and Detection using Deep Convolutional Neural Network*. Penelitian ini menggunakan CNN dengan arsitektur VGG untuk mengklasifikasikan daun jagung yang sehat dan daun jagung yang berpenyakit. Dalam penelitian tersebut juga menggunakan model pembelajaran mendalan yaitu YOLOv4 yang berfungsi untuk mengidentifikasi area daun jagung yang terinfeksi. Dataset yang digunakan berjuamlah 4225 citra dengan resolusi 3456 x 4608 pixel yang terdiri dari du akelas yaitu kelas daun yang sehat dan kelas daun yang terinfeksi. Hasil model klasifikasi yang dihasilkan mencapai akurasi 99,25% dan hasil model deteksi mencapai rata-rata 55,30% (Haque & Adolphs., 2021).

Pada tahun 2020 penelitian yang berjudul *Maize Leaf Disease Detection Using Convolutional Neural Network* mengusulkan peningkatan CNN yang menggabungkan *batch normalization* dan fungsi *central loss* berdasarkan arsitektur VGG16 untuk klasifikasi penyakit daun jagung. Data yang digunakan berupa citra daun jagung sebanyak 4000 citra dengan dibagi menjadi dua kelas yaitu daun jagung sehat dan daun jagung. Model yang diusulkan mendapatkan akurasi pengujian sebanyak 99.2% (Bhanusri dkk., 2020).

Penelitian berjudul *Indentification of Diseases in Corn Leaves using Convolutional Neural Network and Boosting*. Pada penelitian ini membandingkan performa akurasi klasifikasi daun jagung menggunakan

algoritma *Convolutional Neural Network* mengunakan *Boosting* dan tanpa menggunakan *Boosting*. Dataset yang digunakan diambil dari dataset *PlantVillage* yang terdiri dari 4 kelas yaitu *Healthy, Common Rust, Late Blight* dan *Leaf Spot.* Hasil dari perbandingan model *Convolutional Neural Network* tanpa menggunakan *Booster* sebesar 90% sedangkan model yang menggunakan *Booster* sebesar 98% (Bhatt dkk., 2019).

Penelitian lain yang berjudul *Detection of Disease on Corn Plants Using Convolutional Neural Network*, juga menggunakan CNN untuk mengidentifikasi penyakit daun jagung. *Dataset* yang digunakan berasal dari *PlantVillage*. Terdiri dari tiga penyakit daun jagung sejumlah 3.854 citra yang terdiri dari tigas kelas. Metode yang diusulkan mencapai akurasi 99% (Hidayat dkk., 2019).

Berdasarkan referensi penelitian sebelumnya yang dijadikan acuan oleh penulis dalam melakukan penelitian ini. Tujuan dari penelitian ini adalah untuk mengetahui bagaimana algoritma *Deep Learning* dapat mengenali dan mengklasifikasikan citra daun jagung berdasarkan kelas yang telah ditentukan.

2.2 Penyakit Daun Jagung

Tanaman jagung merupakan tanaman pangan utama urutan ketiga setelah padi dan terigu di dunia serta menempati posisi kedua setelah padi di Indonesia (Sudjono, 2018). Dalam kedudukan taksonomi, tanaman jagung tergolong ke dalam keluarga Poaceae dengan nama spesiesnya adalah *Zea mays L* (Muhadjir 2018). Untuk daun sehat pada tanaman jagung, dapat ditunjukkan seperti pada Gambar 2.1.

Gambar 2. 1 Daun Jagung Sehat

Gambar 2.1 merupakan gambar daun jagung yang sehat. Ciri dari daun jagung yang sehat memiliki warna hijau muda atau hijau tua yang tampak segar.

Berikut merupakan penyakit daun jagung yang digunakan pada penelitian ini.

1. Blight

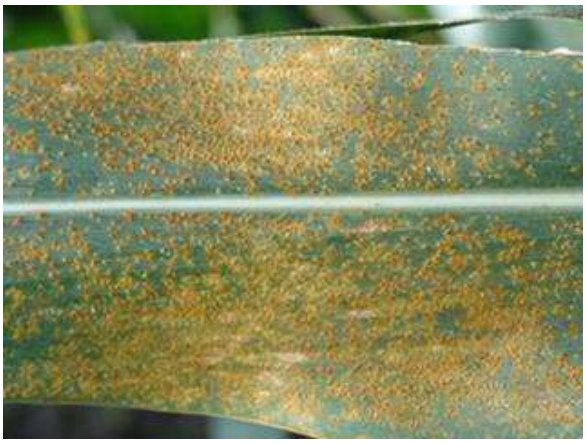
   Blight merupakan penyakit yang menyerang pada tanaman jagung yang menyerang pada bagian daun. Penyakit ini disebabkan oleh jamur H. turcicum. Gejala yang dialami jika terkena penyakit ini adalah terlihat bercak kecil, oval, kebasahan, kemudian bercak memanjang berbentuk elips, menjadi bercak nekrotik (kering) yang luas (hawar), berwarna hijau keabu-abuan atau coklat, dengan Panjang hawar sampai 2,5 sampai 15 cm. Bercak – bercak ini pertama kali terdapat pada daun – daun tua kemudian berkembang menuju daun – daun muda. Bila infeksi cukup berat, tanaman cepat mati, dengan hawar berwarna abu-abu seperti terbakar atau mengering (Sudjono, 2018). Berikut gambar dari daun tanaman jagung yang terkena penyakit Blight, dapat dilihat pada Gambar 2.2.

Gambar 2. 2 Blight

2. Common Rust

   Common Rust merupakan penyakit yang menyerang pada tanaman jagung yang menyerang pada bagian daun. Gejala yang dialami pada tanaman yang terkena penyakit ini terlihat pada permukaan daun atas dan bawah terdapat bercak-bercak kecil (uredinia), bulat sampai oval, berwarna coklat atau merah jingga, Panjang 0,2-2 mm. penyakit ini dapat menyebar ke tanaman jagug

lainnya melalui penyebaran angin (Sudjono, 2018). Penampakan *Commont Rust* ditunjukkan oleh Gambar 2.3.



Gambar 2. 3 Common Rust

3. Grey Leaf Spot

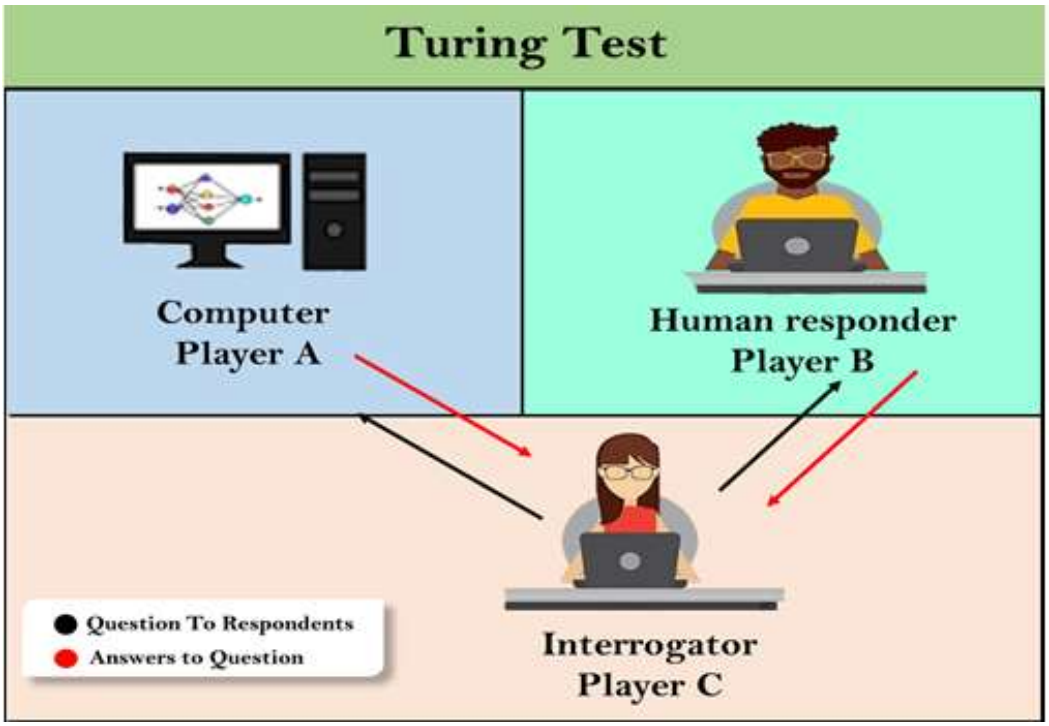
Grey Leaf Spot Rust merupakan penyakit yang menyerang pada tanaman jagung yang menyerang pada bagian daun. Penyakit ini disebabkan oleh jamur *P. maydis*. Gejala yang dialami jika tanaman terkena penyakit ini dapat dilihat pada permukaan dau terdapat garis-garis sejajar tulang daun berwarna putih sampai kuning diikuti dengan garis-garis khlorotik sampai coklat bila infeksi makin lanjut (Sudjono, 2018). Sampel penyakit Grey Leaf Spot ditunjukkan pada Gambar 2.4.

Gambar 2. 4 Grey Leaf Spot

2.3 Artifical Intelligence

Artificial Intelligence atau yang sering disebut kecerdasan buatan merupakan kemampuan mesin untuk meniru penalaran dan kecerdasan manusia

(Berhil dkk, 2019). Pada tahun 1950, Alan Turing seorang ilmuan computer membuat suatu pengujian yang bernama *Turing Test*. *Turing Test* merupakan pengujian yang dilakukan kepada computer untuk mengetahui apakah sebuah komputer dapat berfikir seperti manusia (Russell dan Norvig, 2019).



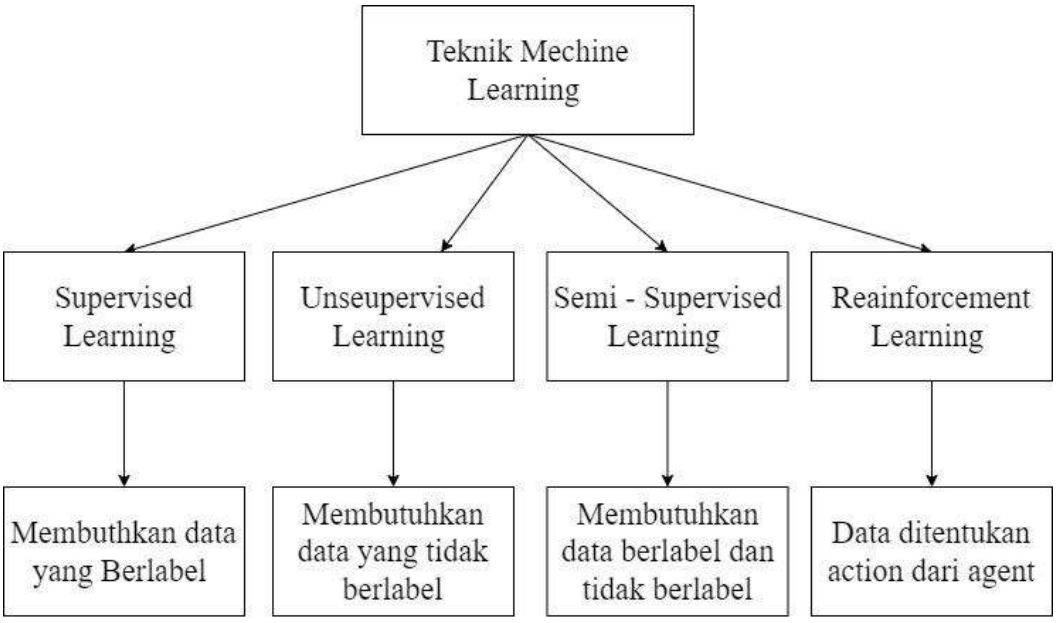
Gambar 2. 5 Pengujian Turing (Javapoint, 2020)

Penjelasan untuk Gambar 2.5, yang merupakan ilustrasi tentang pengujian turning, terdapat tiga pemain dengan seorang akan menjadi pewawancara kemudian kedua lainnya adalah seorang manusia dan komputer sebagai responden. Seorang pewawancara akan memberikan pertanyaan kepada kedua responden dengan kondisi seorang pewawancara tidak mengetahui bahwa salah satunya adalah komputer. Setelah memberikan berberapa pertanyaan kepada keduanya, seorang pewawancara akan melakukan identifikasi terhadap jawaban pertanyaan yang telah diajukan. Kemudian diperintahkan untuk memilih diantara dua responden tersebut yang merupakan manusia dan komputer berdasarkan jawaban pertanyaan keduanya. Apabila pewawancara mengidentifikasi komputer sebagai manusia, maka komputer tersebut berhasil lulus tes dan komputer tersebut memiliki kecerdasan buatan.

2.4 Machine Learning

Machine Learning atau pembelajaran mesin merupakan cabang dari bidang kecerdasan buatan. Perbedaan antara mesin dan manusia adalah kecerdasan, dimana manusia dapat belajar dari pengalaman yang telah dilalui dengan menganalisis data dan membuat keputusan dari pengalaman sebelumnya. Namun Kecerdasan buatan membuat mesin dapat meniru pemikiran seperti manusia. Sehingga membuat mesin dapat deprogram untuk menganalisis dan mengambil keputusan seperti manusia. *Mechine Learning* bertujuan untuk membatu pekerjaan manusia supaya lebih cepat dan lebih akurat dalam menyelesaikan masalah, dan dapat mempelajari data melalui kecerdasan buatan (Samsudiney, 2019).

Mechine Learning memiliki empat teknik pembelajaran yaitu *Supervised Learning, Unsuervised Learning, Semi-supervised Learning, dan Reinforcement Learning*(Makovskaja, 2018).



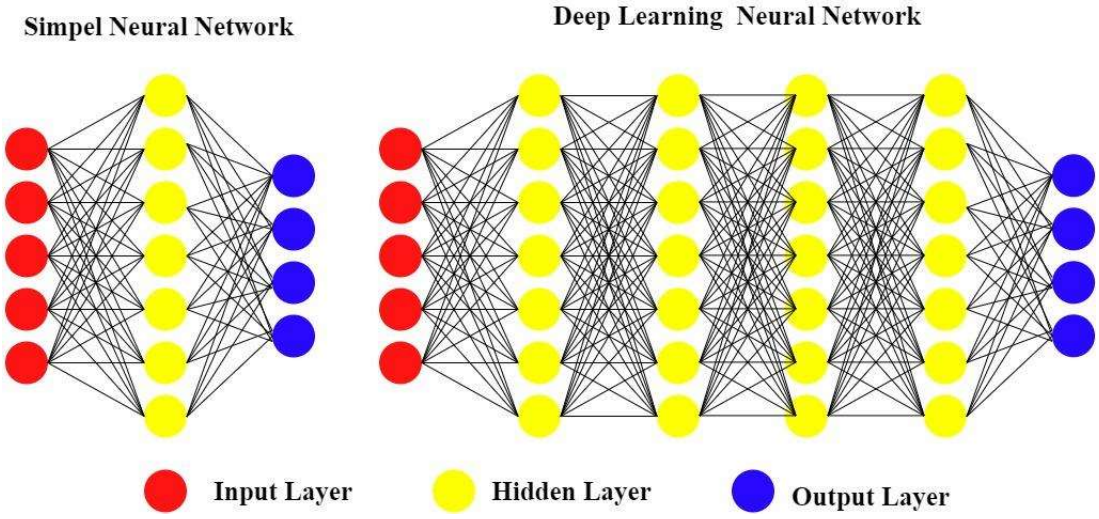
Gambar 2. 6 Teknik Mechine Learning dan Data yang dibutuhkan

Penjelasan untuk Gambar 2.6, *Supervised Learning* atau pembelajaran terawasi secara konsep melatih algoritma pada sekumpulan data berlabel sehingga menghasilkan model pembelajaran yang dapat menyelesaikan masalah berdasarkan data pelatihan yang berlabel. Selain itu, *Unsupervised Learning* secara konsep melatih algoritma pada data yang tidak berlabel, sehingga tujuan model pembelajarannya adalah membuat kelompok atau *cluster* berdasarkan

kesamaan pada data latih. *Semi – Supervised Learning,* metode ini menggabungkan algortima *Supervised* dan *Unsupervised,* dimana data yang diolah terdiri dari data berlabel dan tidak berlabel. Sehingga menghasilkan model yang tepat berdasarjab semua masukan yang diberikan. *Reinforcement Learning* metode ini melatih algoritma berdasarkan konsep "*Reward*" dan " *Punishment*", sehingga model yang dihasilkan dari pembelajaran ini bertujuan untuk mendapatkan "*Reward"* sebanyak banyaknya dan "*Punishment"* sedikit mungkin.

2.5 Deep Learning

Deep Learning merupakan sub bidang dari pengembangan *mechine learning* yang mengajarkan komputer dapat membuat keputusan akurat tanpa bantuan manusia (Grossfeld, 2020). Algoritma *Deep Learning* yang tertanam pada sebuah komputer dapat belajar mengklasifikasi secara langsung dari sebuah data berupa gambar, suara, teks, atau video (LeCun, dkk 2015). Dalam penerapannya, *Deep Learning* menggunakan struktur algoritma jaringan saraf tiruan atau yang sering disebut *Neural Network.* Dalam berberapa permasalahan, *Deep Learning* diterapkan pada macam bidang seperti *Face Recognition, Self – Driving car, Natural Language Processing,* serta lain-lain.



Gambar 2. 7 Ilustrasi arsitektur Deep Learning (Gill,2020)

Pada Gambar 2.7 merupakan Ilustrasi arsitektur dari *Deep Learning*, persamaan antara *Neural Network* biasa dan *Deep Learning Neural Network* ialah diawali menggunakan sebuah *input layer*, diikuti *hidden layer* serta diakhiri menggunakan sebuah output layer. Sedangkan perbedaanya antara keduanya

sangat jelas. Dimana, *Deep Learning Neural Network* mempunyai banyak *hidden layer* daripada *Neural Network* biasa.

2.6 Pengolahan Citra

Pengolahan citra adalah bidang ilmu yang mengkaji perihal bagaimana suatu citra itu dibuat, diolah, serta dianalisis. Sehingga, dapat membentuk informasi baru. Citra sendiri merupakan fungsi asal itensitas cahaya yang direpresentasikan kepada bidang 2 dimensi (Pamungkas, 2017). Citra dalam pembahasan merupakan citra digital yang merupakan gambar hasil proses komputasi asal kamera, *scanner*, serta media digital lainnya. Pengolahan citra berkembang dengan tujuan yaitu: memperbaiki atau menaikkan kualitas tampilan citra (*Image enhancement),* mereduksi ukuran citra (*Image compression),* memperbaiki citra (*Image restoration),* serta mengekstraksi fitur eksklusif dari citra (*Feature extraction)* dan lain-lain (Basuki & Rahmadijanti, 2005). Pengolahan citra terdapat berberapa teknik seperti filtering, transformation, segmentation, masking dan lain-lain. Pada penelitian ini memakai pengolahan citra ditahap praproses data.

2.7 Pencitraan Digital

Pencitraan digital merupakan disiplin ilmu yang mengkaji perihal bagaimana teknik komputasi dari pengolahan sebuah citra. Citra disini merupakan sebuah gambar diam (foto) dan gambar bergerak (video). Sedangkan digital disini adalah pengolahan citra yang dilakukan menggunakan komputer (Sutoyo dkk., 2009). Citra digital dipresentasikan dengan matriks , sehingga pengolahan pada citra digital pada dasarnya memanipulasi elemen-elemen matriks yang berupa piksel yang terdapat pada sebuah larik (A'la, 2016).

Secara matematis, citra merupakan fungsi kontinyu (*continue)* dengan itensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer, maka sebuah citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Proses mempresentasikan citra dengan fungsi kontinyu menjadi diskrit disebut dengan digitalisasi citra. Sebuah citra digital dapat mewakili sebuah matriks dua dimensi f(x,y) yang terdiri dari M kolom dan N baris, dimana

perpotongan antara kolom dan baris disebut piksel (*pixel = picture element)* atau elemen terkecil dari sebuah citra.(Kusmanto dan Tompunu, 2011). Piksel memiliki dua parameter, yaitu koordinat dan itensitas. Nilai yang terdapat pada koordinat (x,y) adalah f(x,y), yaitu besar itensitas dari piksel pada titik tertentu. Sebuah citra dapat dituliskan pada sebuah matriks:

$$f(x,y) = \begin{pmatrix} f(0,0) & f(0,1) & \cdots & f(0,M-1) \\ \vdots & \vdots & & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{pmatrix} \quad (1)$$

Berdasarkan rumus diatas, suatu citra f(x,y) dapat dituliskan ke dalam fungsi matematis seperti berikut :

$$\begin{aligned} 0 &\leq x \leq M-1 \\ 0 &\leq x \leq N-1 \\ 0 &\leq f(x,y) \leq G-1 \end{aligned} \quad (2)$$

Dimana :

M = Jumlah piksel baris pada array citra

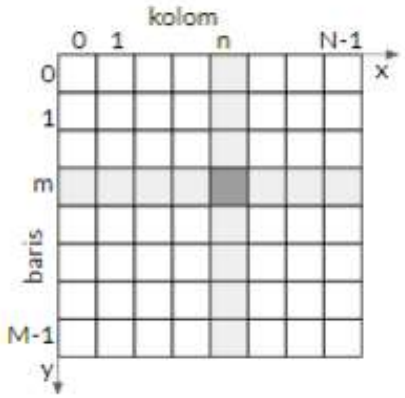
N = Jumlah piksel kolom pada array citra

G = Nilai skala keabuan (*grayscale)*

Besarnya nilai M,N dan G merupakan perpangkatan dari dua seperti yang terlihat pada persamaan berikut :

$$M = 2m; N = 2n; G = 2k \quad (3)$$

Dimana nilai m,n dan k merupakan bilangan positif. Interval (0,G) disebut dangan (*grayscale)* . Nilai G tergantung pada proses digitalisasinya, biasanya nilai keabuan 0 menyatakan itensitas hitam dan 1 menyatakan itensitas putih. Contoh untuk citra 8 bit, dengan nilai G = 2^8 menghasilkan 256 derajat keabuan, seperti pada Gambar 2.8 (Kusmanto dan Tompunu, 2011).

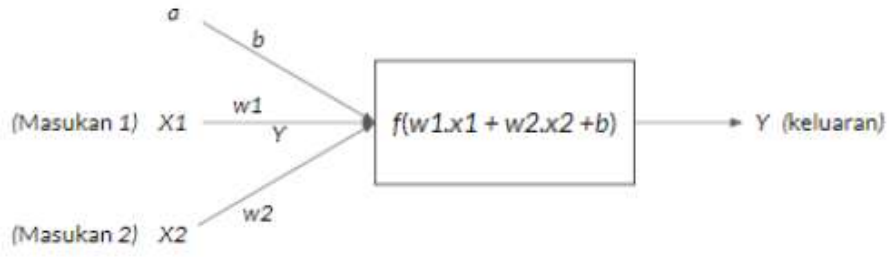


Gambar 2. 8 Citra Digital 2 Dimensi

2.8 Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan ataupun yang lebih dikenal dengan *artificial neural network* merupakan algoritma pembelajaran mesin yang kerap digunakan dan mudah dalam melakukan kustomisasi dengan cukup fleksibel (Fachrurrozi, 2021). Kustomisasi yang diartikan adalah bisa merubah pada arsitektur algoritma semacam menentukan jumlah lapisan, node lapisan dan jenis pada tiap lapisannya sesuai dengan kebutuhan. Versi awalan dari algoritma jaringan syaraf tiruan tersebut bernama *Perceptron* yang diperkenalkan pada sekitar tahun 1958 oleh Rosenblatt (Rosenblatt, 1958).

Perihal yang sangat mendasar pada komputasi jaringan saraf merupakan neuron, ataupun node. Node ini menerima masukan dari node yang lain ataupun dari sumber eksternal yang dihitung buat memperoleh suatu keluaran. Tiap masukkan memiliki bobot (*w*) tertentu, yang mana bobot tersebut diberikan dengan dasar hubungan dengan masukan yang lain. Node tersebut menggunakan sebuah fungsi *f* (yang didefinisikan ke dalam masukan yang sudah diberi bobot(Nisa', 2020).



Gambar 2. 9 Arsitektur sebuah neuron (Nisa', 2020)

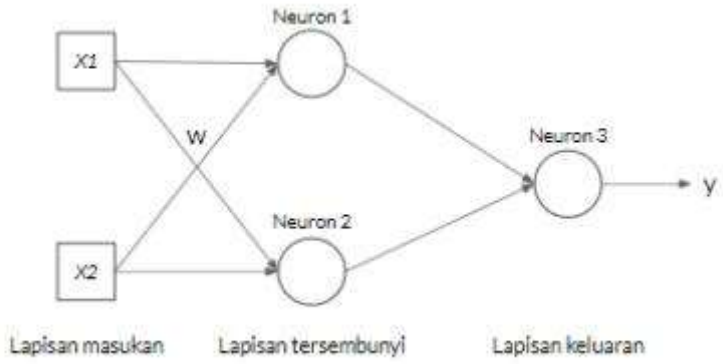
Pada Gambar 2.9 mempresentasikan suatu neuron dengan masukan berupa *x1* serta *x2* dengan bobotnya adalah *w1* serta *w2*. Tidak hanya dari masukan dan bobot, juga terdapat masukan lain berupa *a* dengan bobot ***b*** (bias). Fungsi Utama dari bias merupakan penyedia setiap node dengan sebuah nilai konstan yang dapat dilatih(sebagai tambahan selain dari masukan normal yang diterima) (Nisa', 2020).

Hasil dari neuraon tersebut berupa Y yang merupakan hasil perhitungan dari fungsi *non-linier* yang disebut dengan fungsi aktivasi. Tujuan fungsi aktivasi untuk menunjukkan *non-linier* terhadap keluaran dari neuron. Setiap fungsi aktivasi mengambil sebuah angka dan mengerjakan sebuah operasi matematika yang telah ditentukan didalamnya (Nisa', 2020).

Struktur dari jaringan syaraf tiruan terdiri dari lapisan masukan, lapisan tersembunyi serta lapisan keluaran. Lapisan masukan terhubung dengan satu atau lebih lapisan tersembunyi diikuti dengan nilai bobot tertentu dari lapisan masukan. Lapisan masukan akan memberikan informasi dari luar jaringan. Sehingga pada tahap ini tidak terdapat perhitungan, dikarenakan lapisan tersebut hanya memberikan informasi yang akan diolah kedalam lapisan tersembunyi.

Setelah informasi tersebut masuk ke lapisan tersembunyi, lapisan tersembunyi akan melakukan perhitungan dan akan meneruskan informasi yang telah diolah pada lapisan tersembunyi menuju ke lapisan keluaran. Perlu diketahui bahwa sebuah jaringan tidak hanya memiliki lapisan masukan dan lapisan keluaran, tetapi dapat memiliki lapisan tersembunyi atau tidak.

Setelah melalui proses perhitungan pada lapisan tersembunyi, maka informasi akan diteruskan pada lapisan keluaran, dimana lapisan keluaran akan menampilkan hasil akhir serta menyampaikan informasi yang diperoleh ke luar jaringan. Pada Gambar 2.10. dibawah ini merupakan struktur dari jaringan syaraf tiruan.



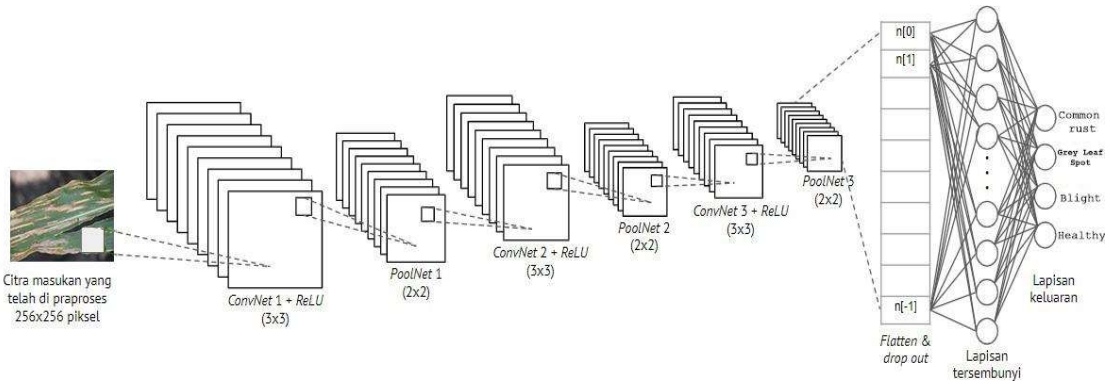
Gambar 2. 10 Struktur Jaringan Syaraf Tiruan (Nisa', 2020)

2.9 Convolutional Neural Network

Convolutional Neural Network merupakan algoritma yang biasanya terkenal digunakan dalam pengenalan objek, klasifikasi citra serta pengelompokan (Shukla & Iriondo, 2020). CNN dikembangkan pertama kali dengan nama *NeoCognitron* oleh Kumihiko Fukushima, seorang peneliti dari NHK *Broadcasting Science Research Laboratories,* Kinuta, Setagaya, Tokyo, Jepang (Fukushima, 2004). Konsep tersebut kemudian dikembangkan oleh seorang peneliti dari *AT&T Laboratories* di Holmdel, New Jersey, USA, yang bernama Yann Lecun dengan arsitektur CNN bernama LeNet yang digunakan dalam melakukan pengenalan pada angka dan tulisan tangan (Lecun, dkk., 1998). Pada riset yang dilakukan oleh Lecun dan kawan- kawan yang terinspirasi oleh penelitian NeoCognitron yang berjudul "Gradien-Based Learning Applied to Document Recognition", membuktikan bahwa model CNN yang menggabungkan fitur yang lebih sederhana dengan fitur yang kompleks mampu mengenali tulisan tangan dengan baik (Lecun dkk., 1998).

Secara teori dapat dikatakan bahwa CNN merupakan bentuk pengembangan terkini dari jaringan syaraf tiruan pada implementasi pengolahan citra yang direpresentasikan pada suatu matriks. Pengembangan yang diartikan ialah peningkatan lapisan konvolusi serta lapisan pooling sebelum data masukan diolah melalui lapisan tersembunyi yang saling terhubung (*fully-connected)*, setelah itu diteruskan ke lapisan luaran. Tetapi ada beberapa perbandingan yang cukup signifikan dari CNN apabila disbanding dengan jaringan syaraf tiruan adalah pada CNN terdapat lapisan konvolusi dan juga adakala terdapat lapisan pooling. Lapisan koncolusi pada dasarnya

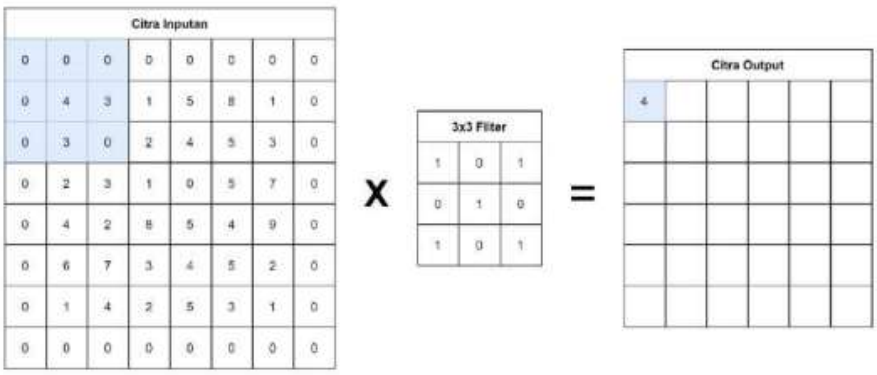
merupakan perkalian pada dua buah matriks, yaitu matriks inputan dengan matriks bobot. Lapisan pooling merupakan salah satu tipe lapisan yang dipakai guna mengurangi dimensi matriks seperti mengambil nilai tertinggi (*max-pooling*), mengambil nilai terendah (*min-pooling*), serta mengambil nilai rata-rata (*average-pooling*). Secara teknis, CNN terdiri dari berberapa tahapan. Masukan (*input)* dan keluaran (*output),*dari setiap tahap terdiri dari berberapa *array* yang bisa disebut *feature map.* Setiap tahapan pada CNN terdiri dari tiga lapisan yaitu lapisan konvolusi, fungsi aktivasi, dan lapisan *pooling*. Berikut merupakan gambar dari jaringan arsitektur CNN.



Gambar 2. 11 Jaringan Arsitektur CNN

2.9.1 Convolution Layer

Lapisan konvolusi merupakan proses utama pada algoritma CNN, Pada proses konvolusi ini pada dasarnya melakukan suatu operasi konvolusi pada matriks kernel sebagaimana pada operasi konvolusi citra pada umumnya (Zhou, 2019). Lapisan konvolusi termasuk dalam bagian dari ekstraksi fitur, lapisan ini bermanfaat guna memperoleh fitur dari suatu citra masukkan dengan memakai suatu penyeleksi khusus dengan menerapkan operasi konvolusi (Yamashita, dkk. 2018).



Gambar 2. 12 Proses Konvolusi (Fachrurrozi, 2021)

Pada Gambar 2.12 merupakan proses konvolusi pada sebuah matriks, Operasi konvolusi sendiri merupakan perkalian dot antara input dan filter yang akan menghasilkan sebuah *feature map* (Brownlee, 2020). Pada Gambar 2.12 terdapat citra inputan dengan ukuran 8 x 8 dan filter atau kernel dengan ukuran 3 x 3. Matriks kernel atau filter ini menjadi acuan perhitungan dalam proses konvolusi, yang akan dilakukan perhitungan perkalian matriks antara matriks citra inputan dan matriks kernel. Pada ilustrasi diatas matriks kernel atau filter akan bergerak dari kiri atas dan akan bergeser ke kanan sebesar *stride* yang ditentukan dan akan bergeser ke bawah dan memulai lagi dari kiri apabila telah mencapai ujung hingga matriks output atau *feature map* terisi penuh (Kohir, 2020). Pada Gambar 2.12, terdapat sebuah citra input berukuran 8 x 8 dengan filter 3 x 3, padding nol (0), dan stirde 1. citra output mendapat nilai 4 didapat dari (1 x 0) + (0 x 0) + (1 x 0) + (0 x 0) + (1 x 4) + (0 x 3) + (1 x 0) + (0 x 3) + (0 x 0) = 4. Rumus perhitungan yang digunakan pada proses konvolusi yaitu :

$$Qj = f\left(\sum_{i=0}^{N} l_{i,j} * k_{i,j} + B_j\right) \quad (4)$$

Keterangan :

Q = Nilai hasil konvolusi

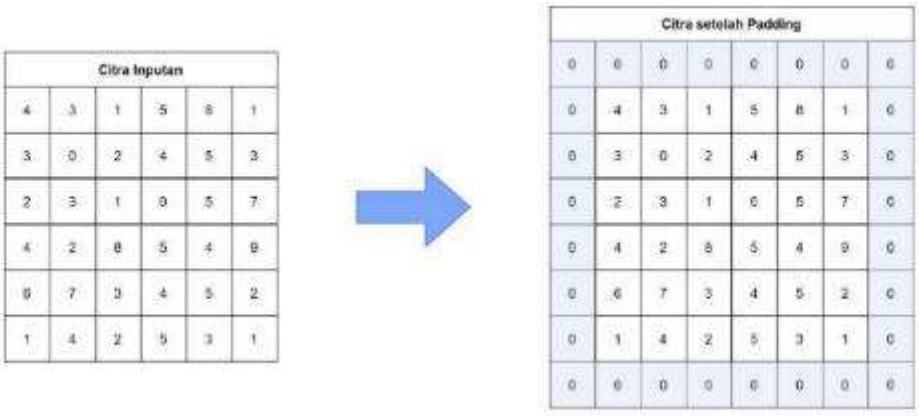
$l_{i,j}$ = Nilai Matriks Citra Inputan

$k_{i,j}$ = Niali Matriks Kernel

B_j = Nilai Bias

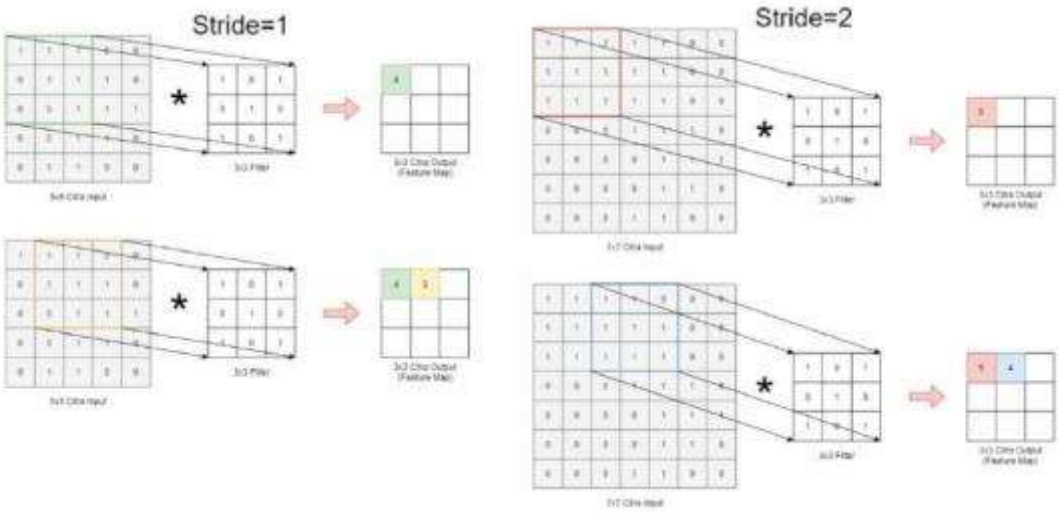
f = Hasil fungsi aktifasi

Padding merupakan istilah pada CNN terkait tentang teknik dalam penambahan jumlah piksel dengan memberikan sebuah nilai yang tetap pada sekitar matriks inputan(riptutorial, 2020). Penggunaan padding diperlukan karena terkadang ukuran matriks kernel yang digunakan tidak sesuai dengan ukuran citra inputan serta tidak mengurangi ukuran citra telah dilakukan konvolusi. Pada Gambar 2.13 melakukan *zero-padding* yaitu menambahkan nilai piksel nol (0) pada sekitar citra inputan. Citra inputan dengan ukuran 6 x 6 setelah diberikan *padding* ukuran citra menjadi 8 x 8.



Gambar 2. 13 Illustrasi Proses Padding (Fachrurrozi, 2021)

Sedangkan *Stride* merupakan teknik yang melakukan pengaturan banyaknya piksel yang akan digeser pada kernel atau filter dalam sebuah citra (DeepAI, 2020). Pada Gambar 2.14 mengillustrasikan bahawa diatur sebanyak yang telah ditentukan, maka kernel atau filter akan bergerak sesuai jumlah yang telah ditentukan.



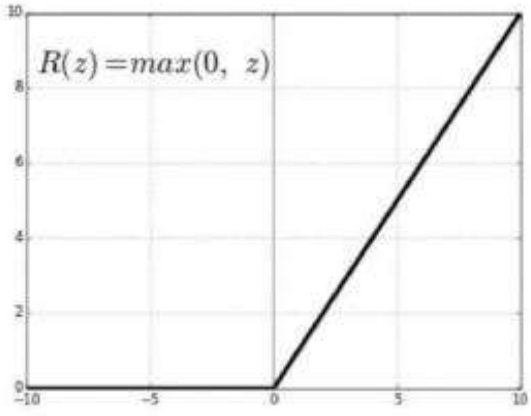
Gambar 2. 14 Illustrasi Proses Stride (Sunu, 2022)

2.9.2 Activation Function Layer

Activation Function Layer atau dalam Bahasa Indonesia merupakan fungsi aktivasi ialai persamaan matematis yang berperan sebai nilai determinan nilai output dari suatu neuron pada neural network (Missinglink, 2020). Fungsi aktivasi memiliki banyak jenis, namun yang sering digunakan dalam pemrosesan yaitu *non-linear functions,* karena memungkinkan model melakukan operasi dalam penanganan yang kompleks pada input dan output suatu jaringan yang

besar (Kapkar, 2020). Berikut jenis fungsi aktivasi *non-linear functions* diantaranya (Nwankpa dkk, 2018) :

1. Rectified Linear Unit (ReLu)



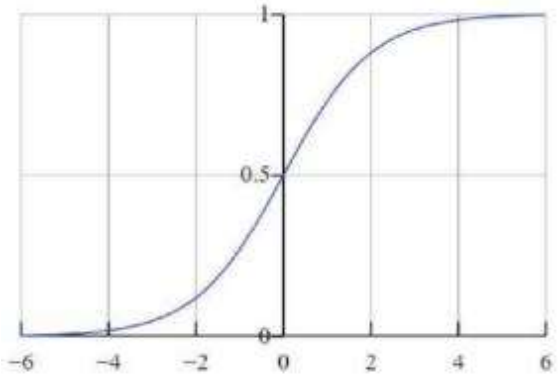
Gambar 2. 15 Grafik Fungsi ReLu

Pada Gambar 2.15 merupakan ilustrasi grafik dari *Rectified Liniear Unit* atau ReLu merupakan fungsi aktivasi *non-linear* yang akan mengkonversi nilai masukan kurang dari 0 (nol) menjadi nilai 0 (nol), dan akan mengeluarkan nilai sama dengan masukkan jika nilai masukkan lebih dari 0 (nol). ReLu memiliki persamaan fungsi sebagai berikut:

$$f(x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases} \quad (5)$$

2. *Sigmoid*

Sigmoid merupakan salah satu fungsi aktivasi yang dipakai untuk pembelajaran mesin paling utama pada aspek *logistic regression,* peranan aktivasi ini kerap dipakai untuk identifikasi pembelajaran mesin, akan tetapi tidak sering digunakan untuk model pembelajaran mesin pada tingkat lanjut (Fachrurrozi, 2021). Illustrasi Grafik Sigmoid dapat dilihat pada Gambar 2.16.



Gambar 2. 16 Grafik Fungsi Sigmoid

Berikut rumus persamaan dari fungsi aktivasi sigmoid :

$$f(x) = \frac{1}{1+e^{-x}} \quad (6)$$

3. *Softmax*

Fungsi *Softmax* merupakan fungsi yang mengganti nilai riil K menjadi vector nilai rill K yang berjumlah 1. Nilai masukan dapat berupa nilai positif, negatif, nol, atau lebih besar dari satu. Namun fungsi *Softmax* mengubahnya jadi angka antara 0 dan 1, alhasil bisa dimaksud sebagai probabilitas. Apabila salah satu masukan kecil (negatif) fungsi *Softmax* mengubahnya jadi probabilitas kecil dan jika masukan besar (positif) fungsi *Softmax* mengubahnya menjadi probabilitas besar, namun nilai bakal selalu antara 0 dan 1 (Wood, 2019).

2.9.3 Pooling Layer

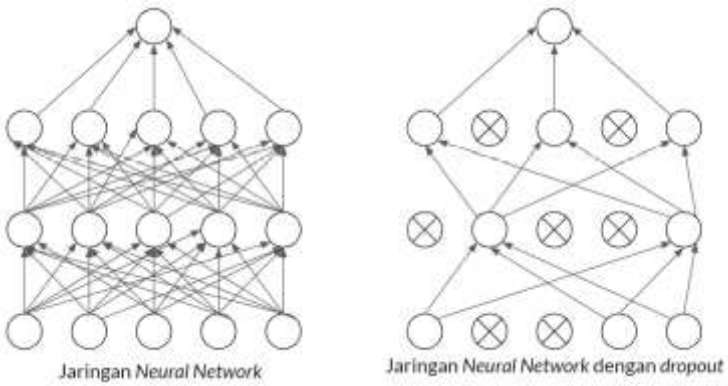
Pooling layer ataupun lapisan *pooling* merupakan suatu lapisan yang biasanya berguna dalam mengurangi sebuah dimensi dari suatu citra dengan mengambil nilai bagian tertentu citra yang berasal dari lapisan konvolusi (Zhou dan Tan, 2020). Lapisan yang terdapat sesudah lapisan konvolusi ini pada dasarnya terdiri dari suatu filter dengan ukuran serta stride tertentu yang akan bergerak pada semua area *feature map* secara berkala serta mengurangi ukuran volume keluaran pada *feature map*. Bentuk lapisan pooling biasanya menggunakan *filter* dengan ukuran 2x2 yang diterapkan dengan melangkah sebanyak dua dan beroperasi pada setiap irisan dari masukkannya. Pada pengambilan nilai elemen tertentu dapat dilakukan dengan tiga cara yaitu mengambil nilai tertinggi (*MaxPooling*), nilai terendah (*MinPooling*) atau nilai rata-rata (*AvaragePooling*) (Zhou, 2019). Berikut illustrasi proses pada *Pooling Layer*, dapat dilihat pada Gambar 2.17.

Gambar 2. 17 Illustrasi Pooling Layer

Pada Gambar 2.17 tersebut terdapat sebuah *feature map* dari sebauh citra yang memiliki ukuran 4x4, setelah melewati proses pooling dengan filter 2x2 dan stride sebesar 2, maka akan menjadi sebuah matriks baru dengan ukuran 2x2. Contoh pooling diatas merupakan jenis *MaxPooling,* mengambil nilai tertinggi pada sebuah citra masukkan.

2.9.4 Dropout Layer

Dropout layer atau yang sering disebut lapisan *dropout* merupakan suatu prosedur yang digunakan dalam menangani *overfitting. Overfitting* terjadi sebab arsitektur yang besar tetapi dilatih dengan dataset yang relatif kecil. Penyesuaian pada seluruh kemungkinan model pada suatu dataset yang serupa, setelah itu mencari nilai rata-rata perkiraan tiap model pembelajaran merupakan salah saru metode menangani *overfiting*. Dalam proses *dropout*, memilih berberapa *neuron* secara acak dan tidak akan dipakai selama proses pelatihan, dengan kata lain *neuron-neuron* tersebut dibuang secara acak. Hal ini berarti bahwa kontribusi *neuron* yang dibuang akan diberhentikan sementara, jaringan dan bobot baru juga tidak diterapkan pada *neuron* pada saat melakukan propagasi balik (Nisa', 2020). Berikut merupakan illustrasi proses *dropout :*

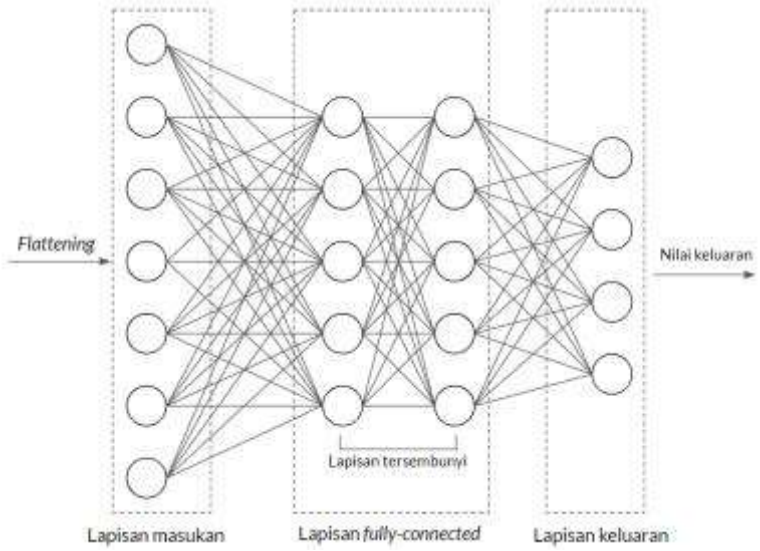


Gambar 2. 18 Illustrasi dropout layer (Srivastava dkk, 2014)

Pada Gambar 2.18 pada bagian kiri merupakan jaringan saraf tiruan tanpa menggunakan *dropout* yang memiliki dua lapisan tersembunyi. Sedangkan pada bagian kanan merupakan jaringan saraf tiruan menggunakan *dropout*. Dari gambar tersebut tampak bahwa jaringan saraf tiruan yang menggunakan *dropout* terdapat berberapa *neuron* aktivasi yang tidak digunakan lagi. Penggunaan teknik ini pada model CNN akan berdampak pada performa model dalam melakukan pelatihan serta mengurangi terjadinya *overfitting* (Srivastava dkk, 2014).

2.9.5 Fully-Connected Layer

Fully-Connected Layer merupakan sautau lapisan dimana seluruh neuron aktivasi dari lapisan sebelumnya tersambung seluruhnya dengan neuron di lapisan berikutnya. Perbandingan antara lapisan *Fully-Connected* dan lapisan konvolusi merupakan neuron di lapisan konvolusi adalah neuron di lapisan konvolusi terhubung hanya ke daerah tertentu pada masukan, sedangkan lapisan *Fully-Connected* mempunyai neuron yang secara keseluruhan tersambung. Gambar 2.19 merupakan illustrasi *Fully-Connected Layer.*



Gambar 2. 19 Illustrasi Fully-Connected Layer

2.9.6 Loss Function

Fungsi ini adalah salah satu bagian dari algoritma bernama *Stochastic Gradient Descent (*SGD) dalam *Deep Learning* yang berfungsi dalam mengestimasi besaran *loss* dari suatu model maka *weight* yang akan didapatkan bakal dilakukan pembaruan guna untuk mengurangi besaran *loss* pada iterasi berikutnya (Brownlee, 2020). Fungsi ini memiliki tiga jenis *loss* yaitu sebagai berikut :

1. *Regression loss functions*

Fungsi loss ini merupakan MSE (*Mean Square Error)*. MSE diukur pada rata rata selisih kuadrat antara prediksi dan data sebelumnya. Rumus MSE dapat dilihat sebagai berikut (Parmar, 2018) :

$$MSE = \frac{\sum_{i=1}^{n}(y - \hat{y})^2}{n} \tag{7}$$

Keterangan :

n = banyak data penelitian

i = data pelatihan ke-i pada data set

yi = nilai actual untuk data pelatihan ke – i

$\hat{y}i$ = prediksi untuk data pelatihan ke-i

2. *Binary classification loss functions*

Fungsi ini biasa dapat digunakan pada penyelesaian terkait masalah biner seperti 0 atau 1 (Peltarion, 2020). Rumus *binary cross entropy :*

$$Loss = -\frac{1}{N}\sum_{i=1}^{N} \gamma_i * \log \hat{y_i} + (1 - \gamma_i) * \log (1 - \hat{y}_i) \tag{8}$$

Keterangan :

yi = nilai actual untuk data ke-i

N = banyak data penelitian

$\hat{y}i$ = niali prediksi untuk data ke-i

3. *Multi class classification loss funtions*

Fungsi ini biasa paling banyak dipakai untuk mengklasifikasi, sebab dapat digunakan pada banyak label kelas. Terdapat dua jenis yang dipakai yakni *Categorical Cross Entropy (CCE)* dan *Sparse Categorical Cross Entropy (SCCE)* (Brownlee, 2020). Perbedaan antara CCE dan SCCE adalah bentuk output yang dihasilkan. SCCE menghasilkan label dengan format integer, sedangkan CCE menghasilkan label dengan format *One-Hot Encoding* (OHE) (Chris,2019).

2.9.7 Optimization Function

Optimization Function ataupun fungsi optimizer adalah fungsi yang dipakai guna untuk memperbarui parameter dari bobot, ini memiliki tujuan agar meminimalisir angka loss yang diterima (Khandelwal, 2019). Fungsi ini masih berhubungan dekat dengan fungsi *loss*. Hal itu disebabkan optimizer memerlukan optimasi pada fungsi loss supaya memperoleh hasil terbaik (Khandelwal, 2019). Dalam *deep learning* terdapat berberapa optimizer, yang kerap dipakai antara lain :

1. *Adaptive Gradient Algorithm (Adagrad)*

Adagrad merupakan suatu metode adaptive learning rate. Pada metode ini learning rate akan dijadikan parameter. Adagard menghilangkan keperluan untuk melakukan adaptasi learning rate secara manual. Tetapi, algoritma optimasi ini memiliki kekurangan yaitu learning rate yang terus menjadi kecil hingga tidak dapat lagi dilatih.

2. *Root Mean Square Propagation (RMSProp)*

Metode ini merupakan penyempurnaan dari kekurangan yang ada pada Adagrad dengan memakai *moving average* dari *gradient* kuadrat. Metode RMSProp memakai besarnya gradient descent terbaru guna menormalkan *gradient*. *Learning rate* pada metode ini deselaraskan dengan cara otomatis serta memakai *learning rate* yang berlainan untuk setiap parameter. RMSProp memisah *learning rate* dengan rata-rata keseluruhan eksponensial dari *gradient descent.*

3. *Adaptive Movement Estimation (Adam)*

Adam merupakan pengembangan dari Adagrad serta kombinasi antara RMSProp dan *Stochastic Gradient Descent (SGD)* dengan momentum (Bushaev, 2018). Metode ini sangat efisien secara komputasi dan dalam prosesnya sedikit memori yang dibutuhkan. Adam optimizer salah satu algoritma *gradient descent* yang sering digunakan.

4. *Stochastic Gradient Descent (SGD)*

Stochastic Gradient Descent biasa disebut sebagai *incremental gradient descent*. Metode ini mencari suatu bobot baru dengan metode pengambilan salah satu data dari semua data training, sesudah itu SGD melakukan Analisa dari setiap data yang diambil. Menfaat memakai SGD yaitu mengurangi penggunaan memori yang diperlukan disaat pemtosesan bobot baru.

2.9.8 Confusion Matriks

Performa suatu model dapat dilihat dari hasil analisa evaluasinya menggunakan *confusion matrix. Confusion matriks* melakukan perhitungan nilai akurasi, recall, precision dan f1 score dari sebuah model pembelajaran mesin (Ghoneim, 2019). Dalam melakukan perhitungan confusion matriks terdapat empat kondisi kasus yang perlu diperhatikan yaitu (Arthana, 2019) :

a. True Positive (TP) : Kondisi saat hasil prediksi benar dan realitanya memang benar.
b. True Negative (TN) : Kondisi saat hasil prediksi benar dan realitanya ternyata salah.
c. False Positive (FP) : Kondisi saat hasil prediksi salah dan realitanya ternyata benar.
d. False Negative (FN) : Kondisi saat hasil prediksi salah dan realitanya memang salah.

Nilai yang didapat dalam kondisi kasus tersebut digunakan untuk menghitung akurasi dengan persamaan berikut :

$$Akurasi = \frac{TP + TN}{TP + FP + FN + TN} \tag{9}$$

Akurasi adalah perbandingan dari data yang telah diprediksi berdasarkan kelas dari keseluruhan data. Untuk menghitung tingkat presisi prediksi kejadian dapat menggunakan persamaan berikut :

$$Presisi = \frac{TP}{TP + FP} \tag{10}$$

Presisi adalah perbandingan data yang sudah dikelompokkan sesuai dengan kelas data yang telah diprediksi dengan kondisi positif. Sedangkan untuk *Recall* dapat dihitung menggunakan persamaan berikut :

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

Recall adalah perbandingan data yang telah terprediksi berdasarkan kelas dengan data yang memang sesuai dengan kelasnya. Untuk F1 Score dapat dihitung menggunakan persamaan berikut :

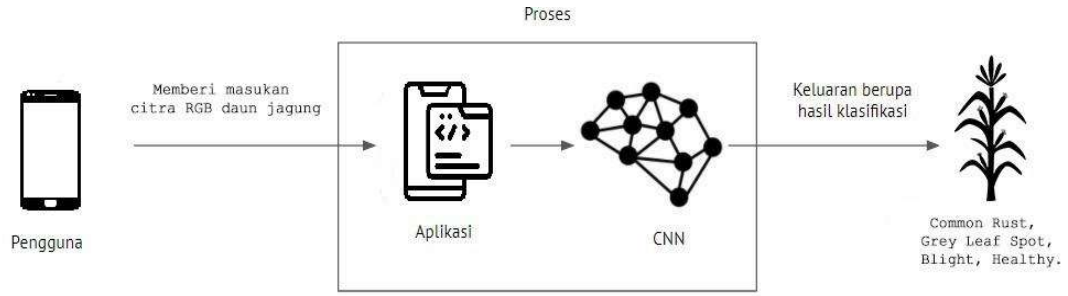
$$F1 - Score = \frac{2\ X\ (Recall\ X\ Presisi)}{(Recall + Presisi)} \quad (12)$$

F1 Score adalah perbandingan rata-rata dari nilai presisi dan nilai recall yang didapat dari proses perhitungan presisi dan recall.

2.10 Deployment Model

Untuk mengetahui berfungsinya sebuah model apakah sesuai dengan yang diharapkan, maka model dapat diimplementasikan melalui bentuk aplikasi agar pengguna dapat mengakses dan berinteraksi dengan model yang telah dibuat (Kawwa, 2019).

Berdasarkah hal tersebut maka rekayasa perangkat lunak pembelajaran mesin dapat digunakan pada aplikasi mobile. Pada Gambar 2.20 Memperlihatkan sebuah sistem pendeteksi penyakit daun jagung sederhana berbasis pembelajaran mesin. Dimana pengguna dapat mengunggah citra berwarna daun jagung kemudian aplikasi melakukan pendeteksian penyakit berdasarkan model algoritma *Convolutional Neuaral Network* berdasarkan citra masukkan, sehingga aplikasi akan mengeluarkan hasil berupa klasifikasi penyakit daun jagung.



Gambar 2. 20 Ilustrasi proses deploy model pada aplikasi

Dikarenakan bidang keahlian yang Penulis pilih dalam penyusunan riset ini tidak berfokus pada bidang rekayasa perangkat lunak melainkan pada sistem cerdasnya sehingga penulis tidak mengulas mengenai perancangan suatu sistem deteksi yang dipakai. Pada hal ini, model yang telah dilatih dan diujicoba akan Penulis tanamkan pada suatu perangkat lunak berupa aplikasi pendeteksi penyakit daun jagung berplatfrom android.