Jurnal Tekno Insentif | Vol. 17 | No. 2 | Oktober 2023
DOI : https://doi.org/10.36787/jti.v17i2.1130 Halaman 112-126

Klasifikasi Penyakit pada Daun Anggur menggunakan Metode *Convolutional Neural Network*

KEN RATRI RETNO WARDANI, LAURENTIUS LEONARDI

Institut Teknologi Harapan Bangsa,Indonesia
Email: ken_ratri@ithb.ac.id

ABSTRAK

Penyakit pada daun dapat mengurangi kualitas dan produktivitas. Deteksi atau klasifikasi penyakit pada tanaman memiliki peran yang sangat menonjol di bidang pertanian modern. Penelitian ini bertujuan untuk mengetahui akurasi klasifikasi objek penyakit pada daun dengan metode *Convolutional Neural Network* dengan menguji ukuran citra, arsitektur, nilai *learning rate,* dan jumlah epoh untuk mendapatkan akurasi yang baik. Selain itu menerapkan regularisasi *Dropout* untuk menghindari *overfitting* dan algoritma optimisasi RMSProp agar laju pembelajaran beradaptasi dengan kondisi data. Dataset yang digunakan berupa citra daun anggur sehat dan berpenyakit, berasal dari PlantVillage. Berdasarkan hasil pengujian, ukuran citra 128 × 128 piksel, arsitektur dengan 5 lapisan konvolusi, nilai *learning rate* 0.001 dan epoh sebanyak 30 menghasilkan nilai akurasi yaitu 98,5% untuk kelas Black Measles, 98,1% untuk kelas Black Rot dan 99,5% untuk kelas Healthy.

Kata kunci: *Convolutional Neural Network*, regularisasi, *Dropout*, *optimizer*, RMSProp, klasifikasi daun

ABSTRACT

Leaf diseases can reduce quality and productivity. Detection or classification of diseases in plants has a very prominent role in modern agriculture. This study aims to determine the classification accuracy of disease objects on leaves using the Convolutional Neural Network method by testing image size, architecture, learning rate, and number of epohs to obtain good accuracy. Besides that, it applies Dropout regularization to avoid overfitting and the RMSProp optimization algorithm so that the learning rate adapts to data conditions. The dataset used is in the form of images of healthy and diseased grape leaves, from PlantVillage. Based on the test results, the image size is 128 × 128 pixels, the architecture with 5 convolution layers, the learning rate is 0.001 and the epoh is 30, which results in an accuracy value of 98.5% for the Black Measles class, 98.1% for the Black Rot class and 99.5 % for Healthy class..

Keywords: *Convolutional Neural Network, regularization, Dropout, optimizer, RMSProp, leaf classification,*

1. PENDAHULUAN

Pertanian modern telah melihat kemajuan besar dalam teknologi untuk mendeteksi penyakit pada tanaman. Kemajuan teknologi pengolahan citra dan kecerdasan buatan digunakan untuk mengidentifikasi penyakit pada tanaman dalam pertanian kontemporer. Jika dilakukan identifikasi secara manual, banyak petani yang gagal mengamati pola penyakit pada daun, hal itu menyebabkan asumsi yang tidak akurat dan merugikan petani **(Kanaka, dkk, 2019)**. Penerapan metode dan teknologi tujuannya untuk menemukan penyakit tanaman dengan lebih akurat dan cepat sehingga memungkinkan pengendalian dan pencegahan penyakit dilakukan lebih dini untuk mengurangi dampak negatif, misalnya hasil panen berkurang.

Banyak penelitian telah dilakukan untuk mengklasifikasi penyakit pada jenis-jenis daun. Salah satunya adalah klasifikasi penyakit pada daun anggur, menggunakan citra daun yang disegmentasi menggunakan *thresholding*, *Gray Level Co-Occurrence Matrix* (GLCM) dan *Radon Transform* untuk mengekstraksi fitur citra, dan metode *Support Vector Machine* (SVM) untuk mendeteksi penyakit. Hasilnya menunjukkan akurasi sebesar 89.90% **(Shweta, dkk, 2018)**. Kelemahan dari metode SVM yaitu memiliki beberapa parameter seperti C (yang mengatur tingkat toleransi terhadap kesalahan) dan kernel yang perlu diatur dengan benar untuk mencapai klasifikasi terbaik dan tidak cocok untuk dataset yang besar.

Perbandingan metode SVM dan *Artificial Neural Network* (ANN) diterapkan untuk klasifikasi beragam penyakit pada daun tomat dan jagung. Tujuannya untuk memprediksi penyakit pada tahap awal sehingga dapat mengambil tindakan pencegahan. Ekstraksi fitur menggunakan *Histogram of Gradient* (HOG). Akurasi terbaik yang diberikan metode SVM 75% sedangkan metode ANN memberikan akurasi terbaik sebesar 85% **(Kanaka, dkk, 2019)**. Dapat disimpulkan bahwa arsitektur ANN yang lebih kompleks dapat menangkap lebih banyak informasi dari data yang rumit dan berstruktur kompleks.

Klasifikasi lainnya menggunakan metode *Convolutional Neural Network* (CNN), dipadukan dengan teknik *Transfer Learning*. Metode ini memiliki keunggulan yaitu dilakukan pra-pelatihan dengan menggunakan informasi yang telah dipelajari dari model CNN yang telah dilatih sebelumnya kemudian mengubah informasi ini untuk tugas klasifikasi baru. Pendekatan ini dapat mengurangi waktu pelatihan dan memerlukan lebih sedikit data untuk mencapai performa yang baik. Kelemahan *Transfer Learning* yaitu *negative transfer*, perlu diingat bahwa *transfer learning* bekerja lebih baik ketika pola-pola data kelas baru sebanding dengan pola-pola yang ada di pra-pelatihan model. Jika pola-pola yang ada di antara kedua tugas sangat berbeda, *transfer learning* mungkin tidak bekerja dengan baik, dan model harus dilatih dari awal menggunakan dataset kelas baru. Keunggulan metode CNN tidak memerlukan *preprocessing*, ekstraksi fitur dan klasifikasi dapat dilakukan sekaligus. Akurasi CNN mencapai 91% untuk mengenali penyakit **(Gulavnai & Patil,2019)**.

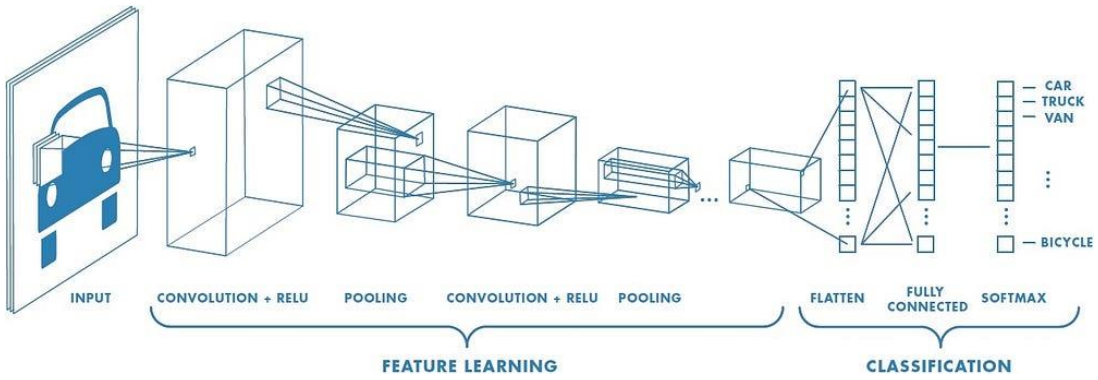
Penelitian lain juga membuktikan kemampuan metode CNN dalam klasifikasi berbagai penyakit pada daun tomat. Penelitian ini hanya melakukan *scale image size* sebagai *preprocessing*. Akurasi terbaik yaitu sebesar 94.66% **(Sangeetha & Mary , 2019)**. Dapat disimpulkan dari beberapa penelitian bahwa metode CNN memiliki kinerja yang sangat baik dibandingkan ANN dan SVM, tidak perlu menggunakan *preprocessing,* untuk klasifikasi data citra. Dengan demikian penelitian ini akan menerapkan metode CNN dengan menguji 3 arsitektur yang berbeda, menguji *hyperparameter learning rate* dan epoh yang optimal, dan mengetahui pengaruh ukuran citra terhadap akurasi. Selain itu menerapkan regularisasi *Dropout* untuk mengurangi *overfitting* dan algoritma optimisasi RMSProp agar laju pembelajaran beradaptasi dengan kondisi data.

2. METODE PENELITIAN

2.1 *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah jenis arsitektur jaringan saraf tiruan yang paling sering diterapkan untuk menganalisis citra visual. Karena kemampuan CNN secara otomatis mengekstraksi elemen penting dari data citra maka CNN sangat efektif dalam pengolahan citra dan *computer vision* **(Venkatesan, dkk, 2017)**. Arsitektur CNN menggunakan lapisan khusus untuk ekstraksi fitur secara hierarkis dari data citra. Konsep kerja CNN memiliki kesamaan dengan *Multilayer Perceptron* (MLP), namun setiap neuron direpresentasikan dalam bentuk dua dimensi.

CNN memiliki dua proses utama yaitu lapisan deteksi fitur dan lapisan klasifikasi. Lapisan deteksi fitur terdapat operasi konvolusi (+*ReLU*) dan *pooling*, lihat gambar 1. Proses ini dilakukan berulang kali pada puluhan atau ratusan lapisan, dan setiap lapisan belajar untuk mengidentifikasi satu atau banyak fitur. Lapisan klasifikasi *Fully connected layer* (FC), menghasilkan vektor dimensi K, di mana K adalah jumlah kelas yang dapat diprediksi jaringan. Probabilitas untuk setiap kelas dari setiap citra yang diklasifikasikan disimpan ke dalam vektor ini. Untuk memberikan keluaran klasifikasi, lapisan terakhir arsitektur CNN menggunakan fungsi *Softmax* **(Salman, dkk, 2018).**



Gambar 1. Arsitektuk CNN

Lapisan konvolusi adalah lapisan yang melakukan operasi konvolusi yaitu kalkulasi perkalian antara citra masukan dan *kernel*/*filter* yang menghasilkan matriks baru berupa peta fitur . *Filter* berupa kisi angka diskrit, memiliki ukuran misalnya panjang 3 piksel dan tinggi 3 piksel, masing-masing elemen merupakan bobot yang diinisialisasi dengan berbagai aturan seperti *Standard Normal Distibution*, *Gaussian Blur*, *Box Blur,* dan lainnya **(Rafael, dkk, 2018)**. Dalam penelitian ini penentuan *kernel* menggunakan *Standard Normal Distibution,* Persamaan (1).

$$k = \frac{1}{\sigma\, x\, \sqrt{2\pi}}\, x\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (1)$$

Keterangan
k : nilai *kernel*
x : variabel *random* normal
μ : *mean*

σ : standard deviasi
π : nilai pi (3.14159265359)
e : bilangan euler (2.718281828459)

Output lapisan konvolusi menghasilkan peta fitur, lihat Persamaan (2), dan terdapat beberapa parameter yang diinisialisasi untuk menghasilkan fitur seperti ukuran *kernel*, *Stride*, *padding*, dan *Output Depth* **(Aniruda, dkk, 2019).**

$$O_{i,j} = \sum_{M=0}^{M-1} \sum_{n=0}^{n-1} I_{(i+m,j+n)} * K_{(m,n)} \quad (2)$$

Keterangan
$O_{i,j}$: nilai matriks keluaran posisi (i, j)
$I_{(i+m, j+n)}$: nilai matriks masukan
Symbol $*$: proses *convolution*
$K_{(m,n)}$: nilai *kernel*
i : koordinat x pada citra
j : koordinat y pada citra
M : lebar citra
N : tinggi citra
m : variabel iterasi untuk lebar citra
n : variabel iterasi untuk tinggi citra

Ukuran citra hasil keluaran dari lapisan konvolusi dihitung menggunakan Persamaan 3.

$$out = \left[\frac{in+2p-k}{s} + 1\right] \quad (3)$$

Keterangan :
out : ukuran matriks keluaran
in : ukuran matriks masukan
p : nilai *padding*
k : nilai *kernel*
s : nilai *stride*

Rectified Linear Unit Layer (ReLU) adalah sebuah fungsi aktivasi untuk mengubah nilai linear menjadi non-linear dengan mengaktifkan dan menonaktifkan neuron **(Le Lu, Zheng, Gustavo, & Lin Yang, 2017)**. Pada persamaan (4), *ReLU* akan memetakan nilai negatif menjadi nol dan mempertahankan nilainya jika positif. Hal ini memungkinkan CNN untuk memodelkan pola-pola yang lebih kompleks dan meningkatkan kemampuan representasi.

$$f(x) = 0 \quad for\ x < 0 \quad (4)$$

$$f(x) = x \quad for\ x \geq 0$$

Lapisan *pooling* digunakan untuk mengurangi dimensi dari peta fitur dengan mereduksi informasi yang tidak penting. *Max pooling* adalah jenis *pooling* yang umum digunakan, di mana nilai maksimum dari area tertentu (2x2) diambil sebagai representasi yang lebih ringkas, Persamaan (5).

$$f(x) = \max(0, x) \quad (5)$$

Lapisan *Fully Connected* mengubah peta fitur menjadi vektor, biasanya ditempatkan di bagian akhir arsitektur. Lapisan ini bertanggung jawab untuk melakukan klasifikasi berdasarkan fitur-fitur yang telah dipelajari sebelumnya. Vektor ini berisi probabilitas untuk setiap kelas dari setiap citra yang diklasifikasikan. *Fully Connected Layer* tujuan untuk melakukan transformasi pada dimensi data yang dapat diklasifikasi secara linear [8]. *Fully Connected Layer* memiliki 1 parameter yaitu *output depth* untuk menentukan panjang keluaran, mewakili setiap nilai kategori hasil klasifikasi. Berikut adalah Persamaan (6) untuk melakukan perhitungan pada *Fully Connected Layer*.

$$O_{(m)} = \sum_{i=0}^{in} K_i^{(m)} \times I_i \quad (6)$$

Keterangan
O(m) : nilai matriks keluaran pada posisi ke-m
K : *kernel*
I : matriks masukan
in : panjang matriks masukan

Lapisan *Softmax* adalah fungsi klasifikasi linier yang digunakan untuk mendapatkan nilai luaran klasifikasi. Berikut Persamaan (7) untuk mendapatkan nilai luaran probabilitas satu objek.

$$f_i(\vec{v}) = \frac{e^{x1}}{\sum_{j=1}^{n} e^{xj}} \quad (7)$$

Keterangan
f_i(v) : probabilitas objek ke-i
e : bilangan euler bernilai 2.71828183
v : vektor nilai seluruh objek
n : panjang v
i : posisi nilai objek

Softmax merupakan lapisan terakhir dalam arsitektur CNN, berfungsi menghitung nilai *loss*. Nilai *loss* ini menunjukkan seberapa besar hasil perhitungan menyimpang dari hasil yang diinginkan dan hasil akhir. Berikut persamaan (8) untuk perhitungan nilai *loss* pada *softmax*.

$$L_i = -\log(f_i(\vec{v})) \quad (8)$$

2.2 Dropout

Lapisan *Dropout* adalah teknik yang digunakan dalam jaringan saraf tiruan, terutama dalam CNN dan *Fully Connected Neural Network* untuk mencegah *overfitting* dan meningkatkan generalisasi model **(Aghdam & Heravi, 2017)(Gangi, dkk, 2021)**. *Overfitting* terjadi ketika model terlalu baik dalam mempelajari data pelatihan, tetapi tidak dapat menggeneralisasi dengan baik pada data baru yang tidak terlihat sebelumnya (data uji atau data dunia nyata), hasilnya akan menjadi buruk. Lapisan *Dropout* bekerja dengan cara menonaktifkan beberapa fitur secara *random* pada saat pelatihan sesuai dengan parameter nilai probabilitas tertentu yang diinisialisasi. Dengan kata lain, parameter atau koneksi yang menghubungkan neuron akan dihapus secara acak selama pelatihan. Jika nilai *random* lebih besar dari nilai probabilitas maka nilai keluaran diubah menjadi nol sedangkan jika nilai *random*

lebih kecil dari *p* maka nilai masukan akan dikalikan dengan 1/*p*. Pada penelitian ini akan menggunakan probabilitas *p* sebesar 0.5 **(Sangeetha & Mary , 2019),** lihat persamaan (9).

$$O_{(i)} = 0 \quad for\ rn > p \tag{9}$$
$$O_{(i)} = I_{(i)}\ x\ \frac{1}{p} \quad for\ rn \leq p$$

Keterangan
$O_{(i)}$: matriks keluaran posisi ke-i
$I_{(i)}$: matriks masukan posisi ke-i
p: nilai probabilitas (0.5)
rn: nilai random dari 0 sampai 1

2.3 *Optimizer* RMSprop

RMSprop adalah algoritma optimisasi yang digunakan dalam pembelajaran mesin untuk memperbarui bobot model jaringan saraf selama proses pelatihan berdasarkan gradien dari *cost function* . *RMSprop* adalah modifikasi dari *Stochastic Gradient Descent* (SGD), algoritma yang membantu mengatasi masalah laju pembelajaran yang mengecil terlalu cepat. RMSprop menggunakan konsep *"moving average"* dari kuadrat gradien untuk menyesuaikan laju pembelajaran untuk setiap parameter model **(Dongpo Xu,Shengdong Zhang,Huisheng Zhang,& Danilo, P., Mandic., 2021).** *Optimizer* membantu algoritma beradaptasi secara adaptif sesuai kondisi data dan mencegah perubahan drastis dalam laju pembelajaran. Nilai yang di rekomendasikan untuk parameter *moving average* atau *beta* adalah 0.9 **(Hinton, Srivastava, & Swersky, 2014).** Persamaan (10), untuk menghitung eksponen dari gradien *moving average*.

$$S_t = \beta S_{(t-1)} + (1-\beta)\left[\frac{\delta L}{\delta Wt}\right]^2 \tag{10}$$

Keterangan
St : Eksponensial *moving average* dari gradien kuadrat
St−1 : Eksponensial *moving average* dari gradien kuadrat sebelumnya
β : Parameter *moving average*
δL/ δWt : Gradien terhadap bobot *kernel*

Kemudian Persamaan (11) untuk melakukan pembaruan bobot *kernel*.

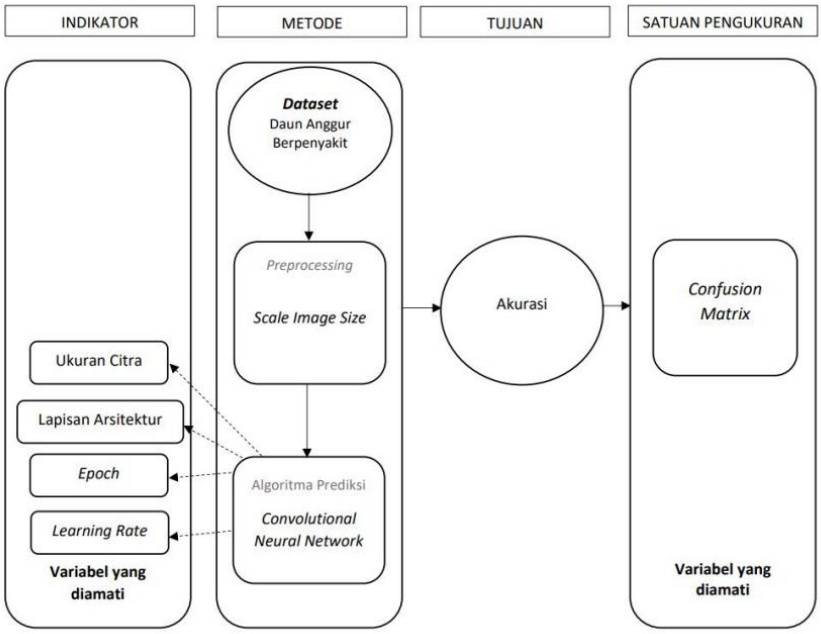
$$W_t + 1 = W_t - \frac{\alpha}{\sqrt{S_t}} \times \frac{\delta L}{\delta Wt} \tag{11}$$

Keterangan :
W_t : Bobot *kernel* asal
W_{t+1} : Bobot *kernel* yang diperbaharui
α : *Learning rate*
S_t : Eksponensial *moving average* dari gradien kuadrat

3. HASIL DAN PEMBAHASAN

Dimulai dengan analisis dataset daun anggur berasal dari PlantVillage **(Ali, 2021)**, pelatihan membuat model CNN, dan pengujian. Kerangka penelitian lihat gambar.2. Evaluasi kinerja

model akan menggunakan *confusion matrix* agar dapat mengidentifikasi jenis kesalahan spesifik yang dilakukan oleh model, seperti situasi di mana model cenderung salah klasifikasi, apakah model lebih cenderung mengabaikan kasus positif atau mengklasifikasikan negatif sebagai positif.



Gambar 2. Kerangka Penelitian

Dataset citra yang digunakan memiliki ukuran awal sebesar 256 × 256 piksel dengan format jpg. Dataset dibagi 2 untuk digunakan sebagai data latih dan data uji. Total data latih adalah 2333 citra dan untuk citra uji sebanyak 670 citra. Dataset dibagi ke dalam 3 kelas yang terdiri dari 2 penyakit daun yaitu *Black Rot* dan *Black Measles*, 1 kelas daun yang sehat atau *Healthy*, lihat gambar 3. Berikut, table 1 keterangan dari dataset tersebut.

Tabel 1. Penjelasan Dataset Daun

Kelas Daun	Deskripsi	Total	
		Citra Latih	Citra Uji
Black Rot	ciri bercak coklat kemerahan dan bintik-bintik berbentuk bulat. Jumlah bintik atau lesi per daun bervariasi dari 2 hingga lebih dari 100 dengan diameter yang berbeda-beda tergantung pada tingkat penyebaran penyakit. Bagian tengah titik daun berubah menjadi kecoklatan dan dikelilingi oleh pinggiran hitam	**966**	**210**
Black Measles atau Esca	ditandai oleh pola *'tiger stripes'* atau strip harimau berwarna coklat kemerahan	**1154**	**240**
Healthy	ciri-ciri daun yang keseluruhan warnanya rata-rata berwarna hijau dan tidak terdapat cacat, bintik maupun area rusak atau berwarna lain	**213**	**220**

Gambar 3. Citra Daun (a) *Black Rot* (b) *Black Measles* (c) *Healthy*

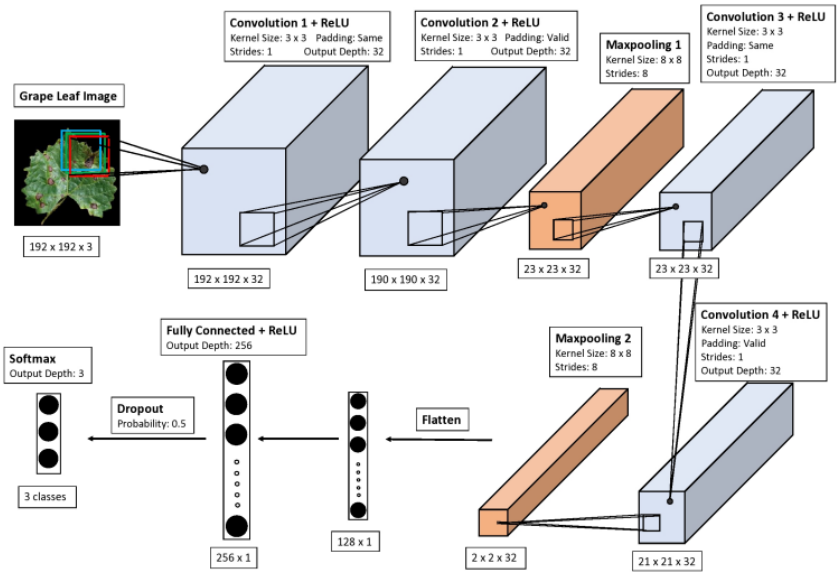
Pengukuran akurasi menggunakan 3 arsitektur CNN, pembeda setiap arsitektur adalah banyaknya lapisan konvolusi. Arsitektur ke-1 memiliki lapisan konvolusi 4 lapis (gambar 4), kemudian arsitektur ke-2 memiliki lapisan konvolusi sebanyak 3 (gambar 5), dan arsitektur ke-3 memiliki lapisan konvolusi sebanyak 5 (gambar 6), rincian lihat tabel 2. *Hyperparameter* yang diuji adalah jumlah epoh, *learning rate*, dan 3 ukuran citra yang berbeda yaitu 256 × 256, 128 × 128, dan 96 × 96 piksel. Nilai uji *learning rate* berkisar 0.0005, 0.00075, dan 0.001. Nilai uji epoh 10,20,30, dan 40.

Tabel 2. Pengujian 3 Arsitektur CNN

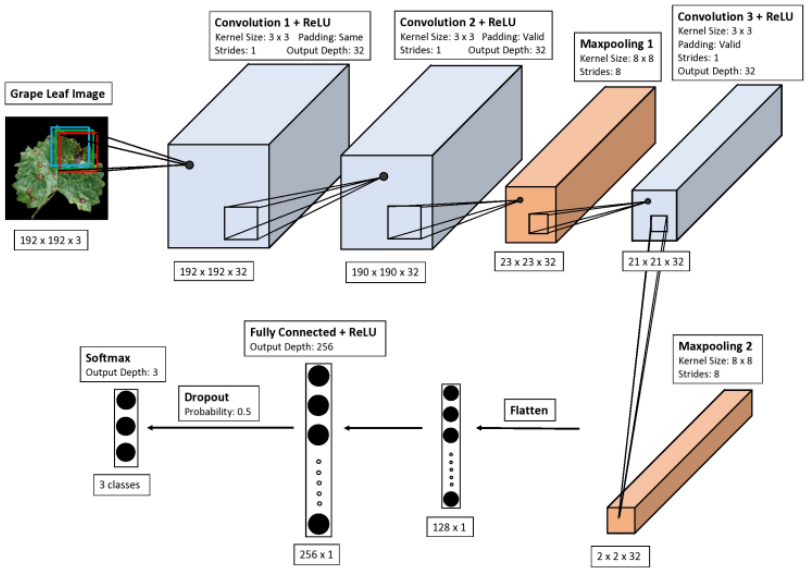
Tahap	Arsitektur 1		Arsitektur 2		Arsitektur 3	
	Lapisan	Parameter	Lapisan	Parameter	Lapisan	Parameter
1	Konvolusi	*Kernel Size: 3* *Padding: Same* *Stride: 1* *Output Depth: 32*	Konvolusi	*Kernel Size: 3* *Padding: Same* *Stride: 1* *Output Depth: 32*	Konvolusi	*Kernel Size: 3* *Padding: Same* *Stride: 1* *Output Depth: 32*
2	ReLU	-	ReLU	-	ReLU	-
3	Konvolusi	*Kernel Size: 3* *Padding: Valid* *Stride: 1* *Output Depth: 32*	Konvolusi	Kernel Size: 3 Padding: Valid Stride: 1 Output Depth: 32	Konvolusi	*Kernel Size: 3* *Padding: Valid* *Stride: 1* *Output Depth: 32*
4	ReLU	-	ReLU	-	ReLU	-
5	*Maxpooling*	*Kernel Size: 8* *Stride: 8*	*Maxpooling*	*Kernel Size: 8* *Stride: 8*	*Maxpooling*	*Kernel Size: 8* *Stride: 8*
6	Konvolusi	*Kernel Size: 3* *Padding: Same* *Stride: 1* *Output Depth: 32*	Konvolusi	*Kernel Size: 3* *Padding: Valid* *Stride: 1* *Output Depth: 32*	Konvolusi	*Kernel Size: 3* *Padding: Same* *Stride: 1* *Output Depth: 32*
7	ReLU	-	ReLU	-	ReLU	-
8	Konvolusi	*Kernel Size: 3* *Padding: Valid* *Stride: 1* *Output Depth: 32*	*Maxpooling*	*Kernel Size: 8* *Stride: 8*	Konvolusi	*Kernel Size: 3* *Padding: Same* *Stride: 1* *Output Depth: 32*
9	ReLU	-	*Flatten*		ReLU	
10	*Maxpooling*	*Kernel Size: 8* *Stride: 8*	*Fully Connected*	*Output Depth: 256*	Konvolusi	*Kernel Size: 3* *Padding: Valid* *Stride: 1* *Output Depth: 32*
11	*Flatten*	-	ReLU	-	ReLU	-
12	*Fully Connected*	*Output Depth: 256*	*Dropout*	*Probabilty: 0.5*	*Maxpooling*	*Kernel Size: 8* *Stride: 8*

Tahap	Arsitektur 1		Arsitektur 2		Arsitektur 3	
	Lapisan	Parameter	Lapisan	Parameter	Lapisan	Parameter
13	ReLU	-	Softmax	*Output Depth: 3*	*Flatten*	-
14	*Dropout*	*Probability: 0.5*			*Fully Connected*	*Output Depth: 256*
15	Softmax	*Output Depth: 3*			ReLU	
16					*Dropout*	*Probability: 0.5*
17					Softmax	*Output Depth: 3*

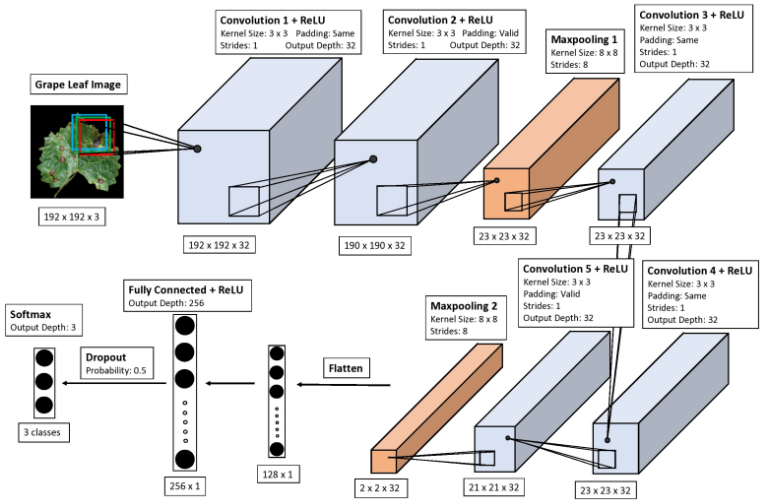
Urutan arsitektur ke 1 lihat Gambar 4, citra masuk melakukan proses konvolusi dengan inisialisasi parameter seperti tabel 2, kemudian diikuti oleh ReLU, proses konvolusi akan diulangi sesuai arsitektur dan diikuti *Maxpooling*. Hasil dari konvolusi akan melalui lapisan *Fully Connected* (*dense*), *Dropout*, dan dilanjutkan dengan *Softmax* untuk menghasilkan nilai probabilitas. Hasil tersebut diteruskan ke proses *backward* menggunakan *optimizer* RMSprop untuk memperbaharui bobot kernel. Method yang digunakan dari pustaka *Tensorflow* adalah tf.keras.*optimizers*.*RMSprop*(), parameter *tuning* yaitu *learning rate* atau *alpha* dengan nilai 0.001 dan rho atau beta dengan nilai 0.9 . Nilai *learning rate* yang optimal akan diuji karena nilai tersebut akan berbeda-beda pada setiap kasus dataset. Penggunaan nilai *learning rate* yang terlalu rendah menyebabkan lambatnya proses pembelajaran untuk mencapai hasil optimal.



Gambar 4. Arsitektur CNN ke-1

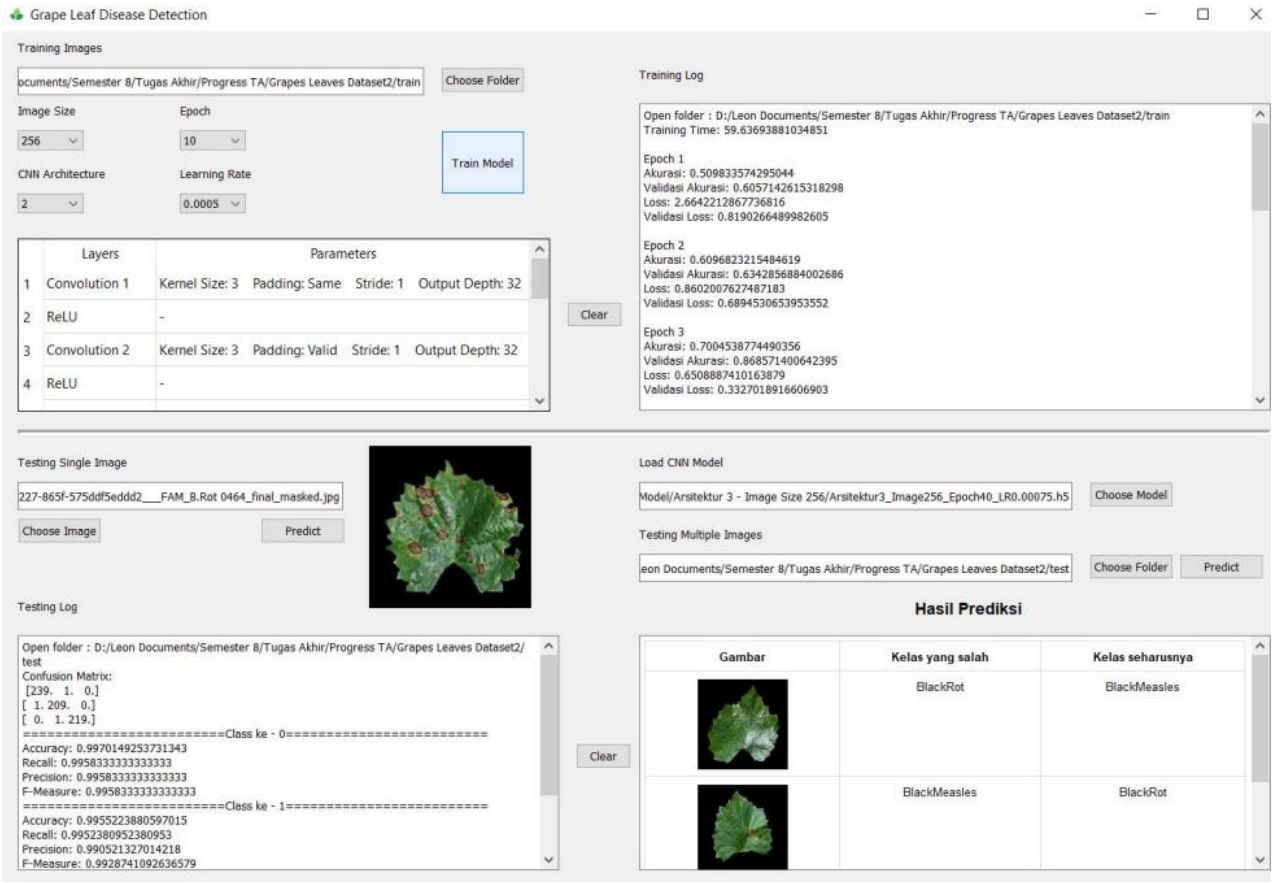


Gambar 5. Arsitektur CNN ke-2



Gambar 6. Arsitektur CNN ke-3

Antarmuka digunakan agar memudahkan pengguna melakukan penyeteman parameter untuk proses latih dan uji, memilih arsitektur, dan melakukan inisialisasi untuk *hyperparameter*, Gambar 7.



Gambar 7. Antarmuka latih dan uji.

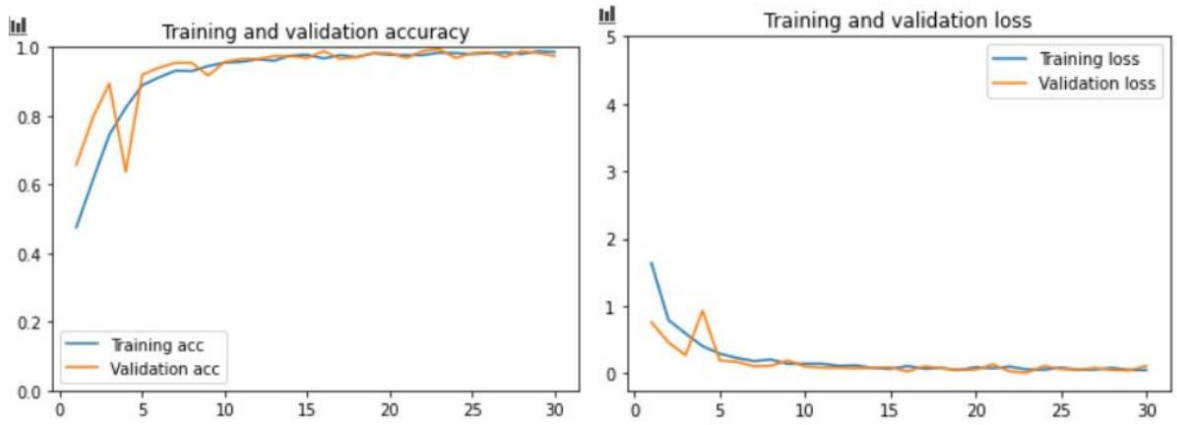
Berikut tabel 3 hasil *Confusion Matrix*, hasil pengujian *multiclass* untuk ke 3 arsitektur, dengan parameter *tuning* terbaik.

Tabel 3. Pengujian 3 Arsitektur CNN

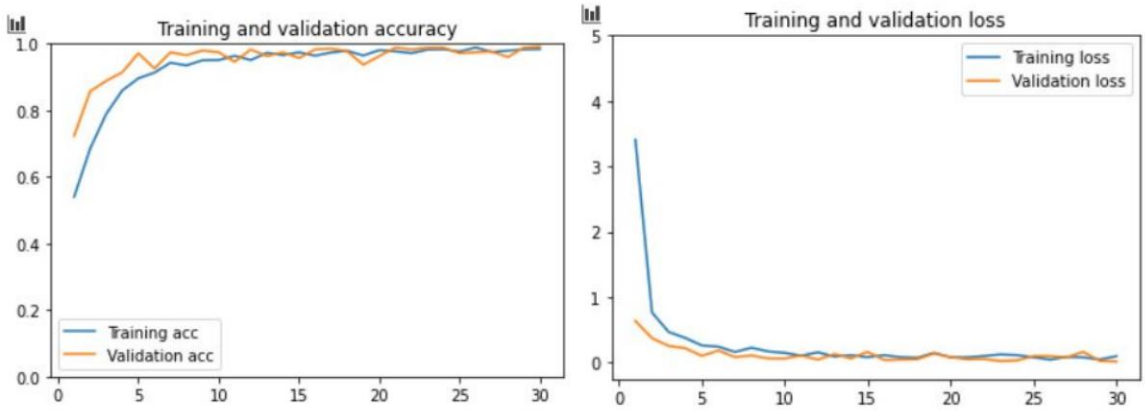
Kelas	Arsitektur	Akurasi	*Recall*	*Precision*	*F-Measure*	Parameter *tuning size*, epoh, *Learning Rate*
Black Measles	1	0.99	0.97	0.99	0.98	192,30, 0.001.
	2	0.99	0.97	0.99	0.98	256, 10, 0.0005
	3	0.993	0.99	0.98	0.99	256,40, 0.0005
Black Rot	1	0.98	0.99	0.94	0.96	192,20, 0.00075
	2	0.98	0.99	0.96	0.98	256,40, 0.001
	3	0.98	0.98	0.98	0.98	256,10, 0.001
Healthy	1	0.99	0.99	1.00	0.99	256,20, 0.001
	2	0.96	0.88	0.99	0.93	256,30, 0.001
	3	0.98	0.97	1.00	0.98	256,20, 0.001

Gambar 8, 9 dan 10 menunjukkan grafik pergerakan nilai akurasi validasi dan nilai *loss* ketika pelatihan yang dihasilkan oleh setiap epoh. Pada gambar, garis berwarna biru menunjukkan pergerakan nilai akurasi dan nilai *loss* untuk data pelatihan sedangkan garis berwarna oranye menunjukkan pergerakan untuk data validasi. Gambar 8, 9 dan 10 menunjukkan ketiga

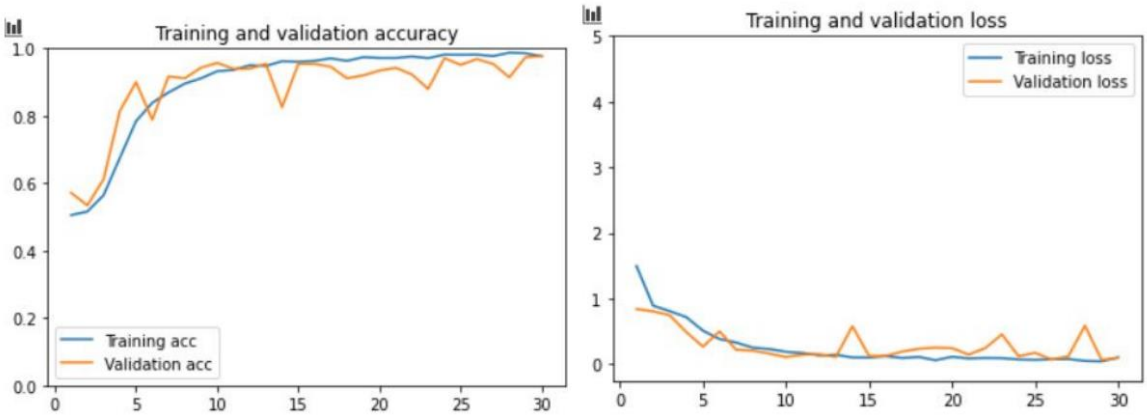
arsitektur mengalami *overfitting* yaitu di mana model menjadi terlalu fokus pada data latih yang digunakan sehingga menangkap data *noise* yang seharusnya diabaikan.



Gambar 8. Akurasi dan nilai *Loss* pelatihan arsitektur ke-1



Gambar 9. Akurasi dan nilai *Loss* pelatihan arsitektur ke-2



Gambar 10. Akurasi dan nilai *Loss* pelatihan arsitektur ke-3

Fluktuasi akurasi pada kurva validasi dan nilai *loss* disebabkan oleh kompleksitas arsitektur yang digunakan. Semakin banyak lapisan yang digunakan maka akan meningkatkan efek fluktuasi. Terbukti pada gambar 10 arsitektur ke-3 yang menggunakan lapisan konvolusi paling

banyak. Secara keseluruhan performa arsitektur ke - 1, 2 dan 3 sudah cukup baik karena kurva validasi akurasi hingga epoh terakhir cenderung mendekati angka 1 hampir sejajar dengan kurva pelatihan dan kurva validasi *loss* terus mendekati angka 0.

Tabel 4 hasil pengujian ukuran citra 128 × 128, 192 × 192 dan 256 × 256 piksel dengan yang optimal *learning rate* 0.001. Ukuran citra tidak memiliki pengaruh yang signifikan terhadap akurasi jika menggunakan nilai *learning rate* yang paling optimal tetapi jika menggunakan *learning rate* 0.0005, 0.005, 0.0025, dan 0.0075 akurasi yang diberikan akan tidak konsisten. Pengaruh paling signifikan semakin besar ukuran citra, waktu pengujian akan jauh lebih lambat.

Tabel 4. Pengujian *Learning Rate* 0.001 dan Epoh 10

Kelas	Arsitektur	Ukuran citra	Akurasi	*Recall*	*Precision*	*F-Measure*	*Time (detik)*
Black Measles	1	128	0.96	0.94	0.96	0.95	28.8
	2	192	0.96	0.91	0.99	0.95	52.10
	3	256	0.97	0.95	0.98	0.96	96.32
Black Rot	1	128	0.95	0.93	0.91	0.92	28.87
	2	192	0.94	0.99	0.85	0.92	52.10
	3	256	0.95	0.97	0.87	0.92	96.32
Healthy	1	128	0.98	0.97	0.973	0.97	28.87
	2	192	0.97	0.93	0.99	0.96	52.10
	3	256	0.97	0.92	0.99	0.95	96.32

Arsitektur ke-3 untuk semua *learning rate* dan epoh menghasilkan akurasi yang paling tinggi karena dipengaruhi oleh jumlah lapisan konvolusi yang lebih banyak yaitu 5 lapisan konvolusi. Semakin banyak jumlah lapisan konvolusi, dapat meningkatkan kemampuan ekstraksi fitur yang lebih banyak sehingga berpotensi menghasilkan nilai akurasi lebih tinggi. Tentunya penggunaan arsitektur dengan lapisan konvolusi yang lebih banyak akan menambah waktu pelatihan.

Berdasarkan hasil pengujian tersebut, nilai *learning rate* yang optimal adalah 0.001 dan memberikan hasil paling konsisten di semua epoh. Nilai *learning rate* tersebut tergolong rendah sehingga memiliki keunggulan yaitu kemampuan belajar menjadi lebih detail. Penggunaan jumlah epoh yang semakin banyak akan memberikan pengaruh yang cukup besar pada waktu pelatihan, tetapi dapat memberikan hasil yang optimal jika nilai *learning rate* lebih kecil.

Hasil pengujian, ukuran citra sebesar 128 × 128 dengan arsitektur ke-3 menghasilkan nilai akurasi yang tinggi dengan waktu pelatihan yang relatif singkat. Nilai *learning rate* sebesar 0.001 dan jumlah *epoh* sebanyak 30 menghasilkan nilai akurasi yang sangat tinggi yaitu 98,5% untuk kelas *Black Measles*, 98,1% untuk kelas *Black Rot* dan 99,5% untuk kelas *Healthy*.

Pada pengujian ini dilakukan analisis kesalahan menggunakan model pelatihan dengan skenario terbaik yaitu arsitektur *layer* ke-3, nilai *learning rate* sebesar 0.001, jumlah epoh sebanyak 30 dan ukuran citra sebesar 128 × 128 piksel. Klasifikasi yang dilakukan menggunakan citra pengujian sebanyak 90 citra yang terdiri dari 3 kelas dengan masing-masing jumlah 30 citra. Analisis ini menunjukkan beberapa faktor yang mempengaruhi

kesalahan klasifikasi misalnya disebabkan terdapat pantulan cahaya pada citra sehingga mengakibatkan kesalahan dalam pengambilan fitur. Kondisi citra yang memiliki kontras yang tidak normal dapat menyebabkan kesalahan klasifikasi.

4. KESIMPULAN

Dari hasil pengujian, dapat ditarik kesimpulan bahwa ukuran citra kurang berpengaruh secara signifikan terhadap akurasi jika menerapkan parameter *learning rate* yang paling optimal yaitu 0.001. Ukuran citra yang besar 256x256 piksel memberikan akurasi yang lebih stabil untuk semua epoh (10, 20, 30, 40) dengan *learning rate* 0.0005, 0.001, dan 0.0075. Arsitektur ke 3 memberikan nilai akurasi yang lebih konsisten untuk semua ukuran citra karena penggunaan jumlah lapisan konvolusi yang lebih banyak pada arsitektur CNN akan meningkatkan kemampuan ekstraksi fitur. Bisa disimpulkan bahwa pada penelitian ini diperlukan model dengan kemampuan belajar yang lebih detail agar dapat melakukan klasifikasi dengan baik sehingga mencapai nilai akurasi yang tinggi. Penggunaan *learning rate* yang semakin rendah perlu diimbangi dengan jumlah epoh yang lebih besar, akan tetapi menyebabkan waktu pelatihan bertambah lama.

DAFTAR RUJUKAN

Aghdam, H. Habibi, & E.,Jahani, Heravi. (2017). *Guide to Convolutional Neural Networks*. Switzerland: Springer.

Ali, Abdalah. (2021, Februari). PlantVillage Dataset. Retrieved from https://www.kaggle.com.

Dongpo Xu,Shengdong Zhang,Huisheng Zhang,& Danilo, P., Mandic. (2021). Convergence of the RMSProp Deep Learning Method with Penalty for Nonconvex Optimization. *Journal Neural Networks Elsevier, 139*, 17-23.

Gulavnai, Sampada, & Patil, Rajashri. Deep Learning for Image Based Mango Leaf Disease Detection". (2019). *International Journal of Recent Technology and Engineering*, *8*, 54 – 56.

Goodfellow, I. , Bengio, I. , & A. Courville. (2016) . *Deep Learning*. 1st ed., England: The MIT Press Cambridge.

Ghosh, Aniruda ,et al. (2019). *Fundamental Concepts of Convolutional Neural Network, in book: Recent Trends and Advances in Artificial Intelligence and Internet of Things*. Switzerland: Springer. 519-567,.

Hinton,G., N., Srivastava, & K., Swersky. (2012). Rmsprop: Divide the gradient by a running average of its recent magnitude. University of Toronto: Department of Computer Science. Canada.

Gangi, Siva Nandini,Siva Kumar, A.P., & Chidananda, K. (2021). Dropout technique for image classification based on extreme learning machine. *Global Transitions Proceedings Elsevier,* (pp 111 – 116).

Kanaka, N., Durga, & Anuradha. (2019). Plant Disease Identification Using SVM and ANN Algorithms. *International Journal of Recent Technology and Engineering*, *7*(554), 471-473

Rafael ,C., Gonzalez, & Richard, E., Woods. (2018). Digital Image Processing. Pearson Education Limited 2018.

Sangeetha, R., & Mary, Shanthi, Rani, R. (2019). Tomato Leaf Disease Prediction Using Convolutional Neural Network. *International Journal of Innovative Technology and Exploring Engineering*, *9*(1), 1348 -1352.

Shalman, Khan., etAll. (2018). *A Guide to Convolutional Neural Networks for Computer Vision.* Morgan & Claypool Publishers.

Le Lu, Yefeng Zheng, Gustavo Carneiro, & Lin Yang. (2017). *Deep Learning and Convolutional Neural Networks for Medical Image Computing*. Switzerland: Springer.

Shweta, S., Kothawale, S.R., Barbade, & Pradnya, P., Mirajkar. (2018). Grape Leaf Disease Detection Using SVM Classifier. *International Journal of Innovative Research in Computer and Communication Engineering*, *6*(4), 3288 - 3295.

Stanford University. CS231n Convolutional Neural Networks for Visual Recognition. CS231n [Online], Available: cs231n.github.io/convolutional-networks/, [Accessed: 3- May-2020].

Venkatesan, Ragav, & Baoxin Li. (2017). *Convolutional Neural Networks in Visual Computing*. London: Taylor & Francis Group, CRC Press.