

BAB 4

IMPELENTASI DAN PENGUJIAN

4.1 Implementasi

Tahap implementasi merupakan tahap pembangunan perangkat lunak pengenalan pola citra sandi rumput sesuai dengan analisis dan perancangan yang sudah dilakukan pada tahap sebelumnya. Tahap ini merupakan tahap dimana perangkat lunak siap untuk digunakan, yang terdiri dari pembahasan mengenai lingkungan implementasi, baik lingkungan perangkat keras maupun perangkat lunak. Berikut merupakan hasil implementasi yang dilakukan.

4.1.1 Implementasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam pembangunan perangkat lunak pengenalan pola citra sandi rumput dengan CNN dapat dilihat pada Tabel 4.1.

Tabel 4.1 Spesifikasi Perangkat Keras

No	Perangkat	Spesifikasi
1	Processor	Intel Core i3
2	Memory	8 GB
3	Harddisk	250 GB
4	Kartu Grafis	Nvidia GeForce 920MX 2 GB

4.1.2 Implementasi Perangkat Lunak

Adapun spesifikasi perangkat lunak yang digunakan dalam pembangunan perangkat lunak pengenalan pola citra sandi rumput dengan CNN dapat dilihat pada Tabel 4.2.

Tabel 4.2 Spesifikasi Perangkat Lunak

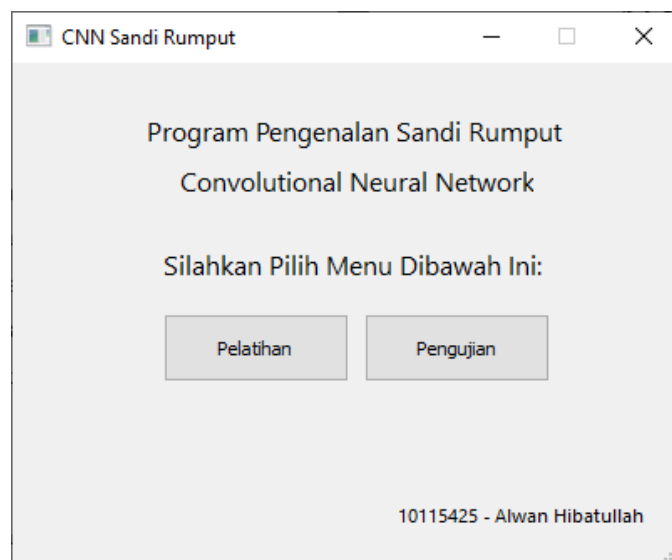
No	Perangkat	Spesifikasi
1	Sistem Operasi	Windows 10
2	Tools Pemrograman	Pycharm 2019.1.3
3	Bahasa Pemrograman	Python 3.6.8

4.1.3 Implementasi Antarmuka

Berikut merupakan implementasi dari tampilan antarmuka pada perangkat lunak yang dibangun.

1. Tampilan Antarmuka Menu Utama

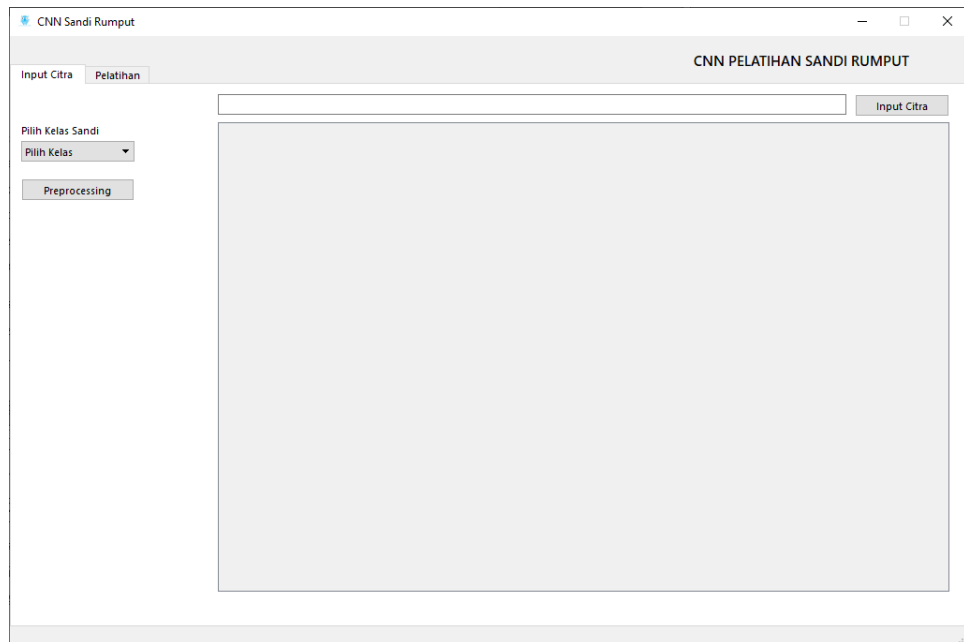
Pada menu utama, terdapat dua tombol yaitu tombol Pelatihan dan Pengujian. Tombol Pelatihan berfungsi untuk masuk ke menu pelatihan sedangkan tombol Pengujian untuk masuk ke menu pengujian. Adapun tampilan antarmuka Menu Utama dapat dilihat pada Gambar 4.1



Gambar 4.1 Tampilan Antar Muka Menu Utama

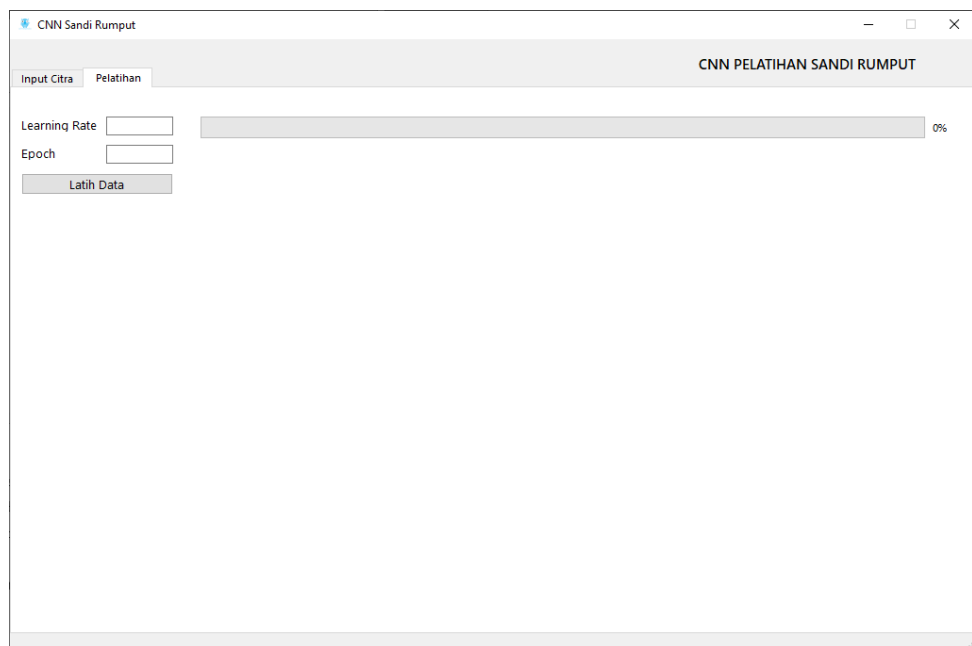
2. Tampilan Antarmuka Menu Pelatihan

Pada menu pelatihan, terdapat dua tab yaitu tab Input citra dan Pelatihan, di dalam tab Input Citra terdapat form untuk memasukkan file citra sandi rumput dan form untuk menampilkan citra yang dimasukkan. Kemudian terdapat combo box untuk memilih kelas sandi yang akan diproses dan terdapat tombol Preprocessing yang berfungsi untuk menjalankan proses pengolahan citra. Adapun tampilannya dapat dilihat pada Gambar 4.2.



Gambar 4.2 Tampilan Antar Muka Menu Pelatihan (Tab Input Citra)

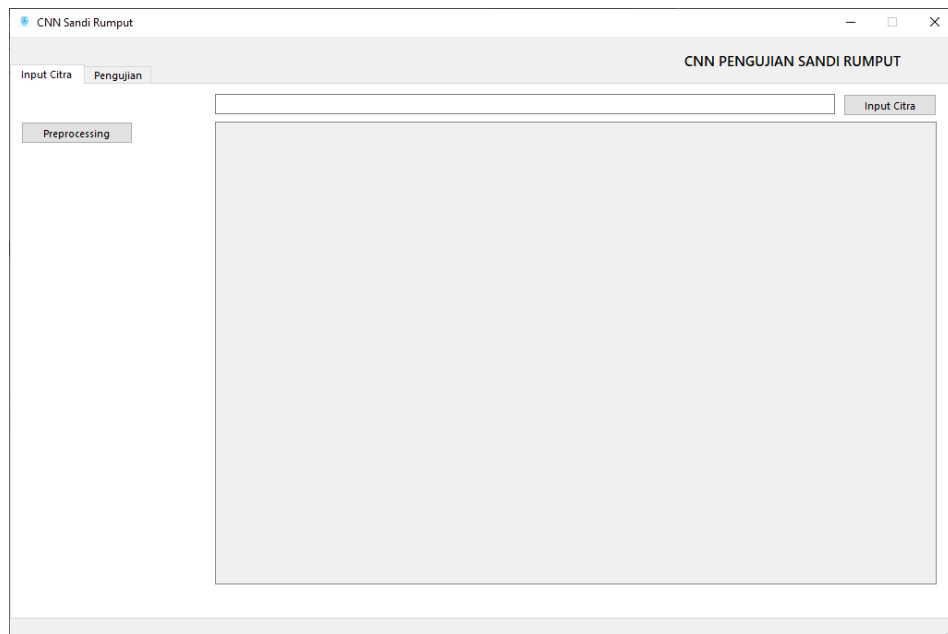
Sedangkan pada tab Pelatihan terdapat form Learning Rate dan Epoch dan tombol Latih data untuk memulai proses pelatihan data. Adapun tampilannya dapat dilihat pada Gambar 4.3



Gambar 4.3 Tampilan Antar Muka Menu Pelatihan (Tab Pelatihan)

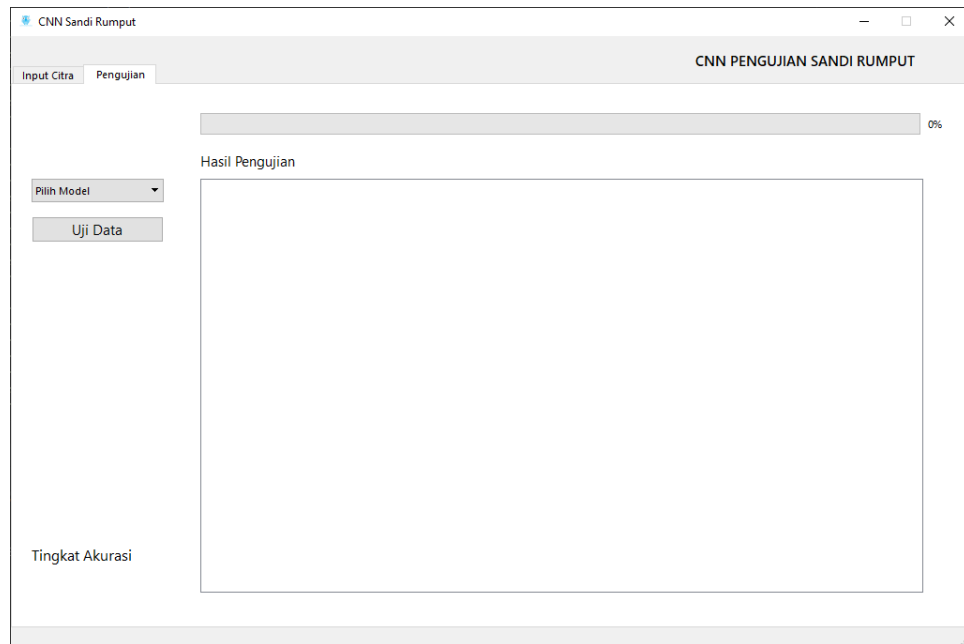
3. Tampilan Antarmuka Menu Pengujian

Pada menu pengujian, terdapat dua tab yaitu tab Input citra dan Pengujian, di dalam tab Input Citra terdapat form untuk memasukkan file citra sandi rumput dan form untuk menampilkan citra yang dimasukkan. Kemudian terdapat tombol Preprocessing yang berfungsi untuk menjalankan proses pengolahan citra. Adapun tampilannya dapat dilihat pada Gambar 4.4



Gambar 4.4 Tampilan Antar Muka Menu Pengujian (Tab Input Citra)

Sedangkan pada tab Pengujian terdapat tabel untuk menampilkan hasil klasifikasi dari citra yang sudah dimasukkan dan tombol Uji Data untuk memulai proses pengujian data serta *combo box* yang berisi model pelatihan. Adapun tampilannya dapat dilihat pada Gambar 4.5



Gambar 4.5 Tampilan Antar Muka Menu Penguujian (Tab Penguujian)

4.2 Penguujian

Penguujian merupakan tahap yang bertujuan untuk menemukan kesalahan-kesalahan atau kekurangan yang ada pada sistem yang sudah dibangun sebelum digunakan oleh pengguna. Tahap penguujian yang dilakukan meliputi penguujian *white box*, penguujian *black box* dan penguujian akurasi.

4.2.1 Penguujian *White Box*

Penguujian Whitebox digunakan untuk meyakinkan semua perintah dan kondisi pada sistem dieksekusi secara minimal. Adapun penguujian whiteboxnya adalah sebagai berikut.

4.2.1.1 Penguujian *White Box* Proses Pelatihan

Berikut ini merupakan *source code* dari proses pelatihan

```

1 load_train = LoadData()
2 if self.txtLR.text() == "" or self.txtEpoch.text() == "":
3     self.messageInfo("Peringatan", "Isi Learning Rate atau Epoch
4         terlebih dahulu!")
5 else:
6     self.data_train = load_train.data_train
7     self.train_generator, self.x_train, self.x_valid, self.y_train,
8     self.y_valid = load_train.loadDataTrain()
9     self.input_shape = (load_train.img_rows, load_train.img_cols,
10        1)
11     epo = self.txtEpoch.text()
12     epo = int(epo)

```

```

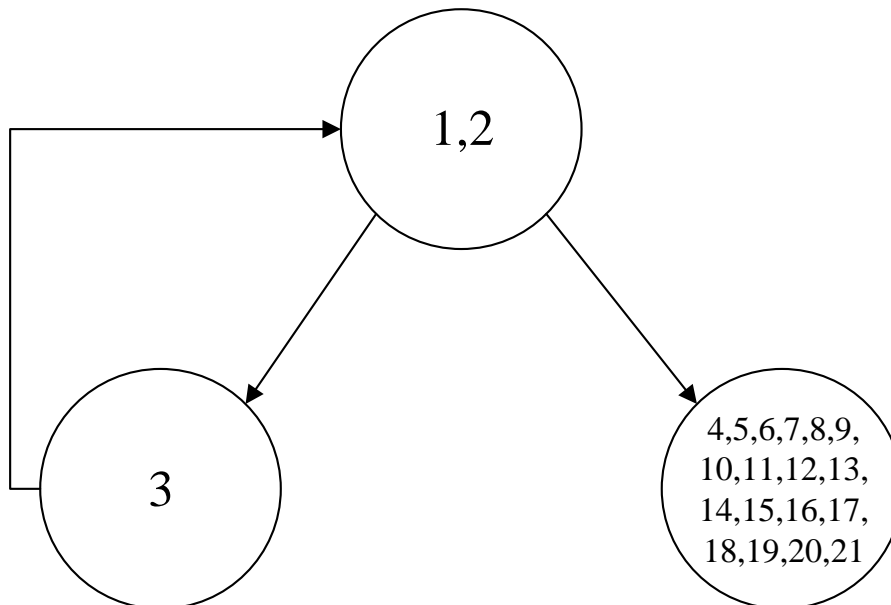
8      lr = self.txtLR.text()
9      lr = float(lr)
10     batch_size = 32
11     pat = int(epo*0.6)
12     call = [EarlyStopping(monitor='val_loss',
13                          min_delta=0.01,patience=pat,verbose=1, mode='min'),
14             ModelCheckpoint('self.txtLR.text()+'.hdf5',
15                             monitor='val_loss', verbose=1, save_best_only=True,
16                             mode='min')
17         ]
13     cnn = CNN(self.input_shape,
14              len(self.train_generator.class_indices), lr)
14     model = cnn.ConvNetModel()
15     self.progressBar.setMinimum(0)
16     self.progressBar.setMaximum(epo)
17     history = model.fit(self.x_train, self.y_train,
18                        batch_size=batch_size, epochs=epo, verbose=1,
19                        validation_data=(self.x_valid, self.y_valid),
20                        callbacks=call)
18     self.progressBar.setValue(epo)
19     evaluation = model.evaluate(self.x_valid, self.y_valid,
20                               batch_size=batch_size, verbose=1)
20     print('loss : %.2f, accuracy : %.2f' % (evaluation[0],
21                                             evaluation[1]))
21     self.messageInfo("Sukses", "Proses Pelatihan Selesai!")

```

Gambar 4.6 Source Code Proses Pelatihan

a. Flow Graph

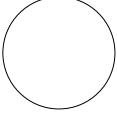
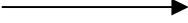
Berikut ini merupakan *Flow Graph* berdasarkan *source code* pada tahap sebelumnya yang digambarkan pada Gambar 4.7



Gambar 4.7 Flow Graph Proses Pelatihan

Keterangan pada Gambar 4.7 dijelaskan pada Tabel 4.3 sebagai berikut.

Tabel 4.3 Keterangan *Flow Graph*

Simbol	Keterangan
	<i>Node (N)</i>
	<i>Edge (E)</i>

Berdasarkan *Flow Graph* pada Gambar 4.7 maka perhitungan *cyclometric complexity* untuk proses pelatihan adalah sebagai berikut.

$$\begin{aligned} V(G) &= E - N + 2 \\ &= 3 - 3 + 2 = 2 \end{aligned}$$

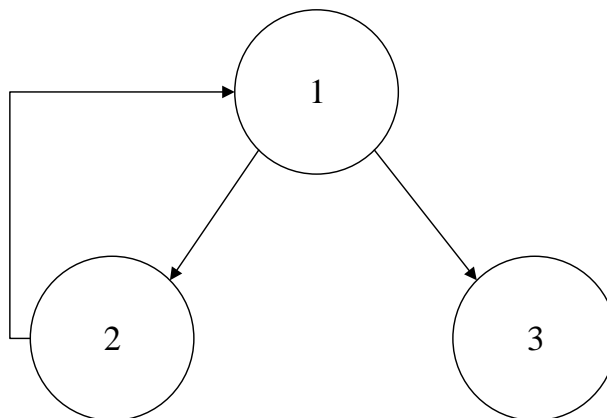
Maka, jumlah *cyclometric complexity* pada proses pelatihan adalah 2. Berdasarkan hasil tersebut, maka terdapat 2 *Independent Path* yang terdiri dari:

Path 1 = 1,2,3,1,2,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21

Path 2 = 1,2,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21

b. Graph Matrix

Agar dapat menghitung *Graph Matrix*, langkah selanjutnya yaitu mengeneralisasikan *Flow Graph* sebagai berikut

**Gambar 4.8 Hasil Generalisasi *Flow Graph* Proses Pelatihan**

Berdasarkan hasil generalisasi *Flow Graph* pada Gambar 4.8 maka *graph matrix* yang didapat sebagai berikut.

Tabel 4.4 Graph Matrix Proses Pelatihan

Node	1	2	3	Graph Matrix
1		1		0
2	1		1	1
3				0
Jumlah + 1				1

4.2.1.2 Pengujian *White Box* Proses Pengujian

Berikut ini merupakan *source code* dari proses pelatihan

```

1  idx = self.cbLR.currentIndex()
2  modelFile = str(self.cbLR.currentText())
3  if idx == 0:
4      self.messageInfo("Peringatan", "Silahkan Pilih Model
5      Terlebih Dahulu!")
6  else:
7      load_test = LoadData()
8      self.folder_test = load_test.data_test
9      self.train_generator, self.x_train, self.x_valid,
10     self.y_train, self.y_valid = load_test.loadDataTrain()
11     self.test_generator, self.x_test =
12     load_test.loadDataTest(self.folder_test)
13     model = load_model('./Model/'+modelFile)
14     self.test_generator.reset()
15     pred = model.predict_generator(self.test_generator,
16     verbose=1, steps=2600/10)
17     self.progressBar.setValue(50)
18     predicted_class_indices = np.argmax(pred, axis=1)
19     labels = (self.train_generator.class_indices)
20     labels = dict((v,k) for k, v in labels.items())
21     prediksi = [labels[k] for k in predicted_class_indices]
22     path = self.test_generator.filename
23     filenames = []
24     for x in range(len(path)):
25         filenames.append(path[x][2])
26     true_pred = 0
27     compare = []
28     for x in range(len(filenames)):
29         if filenames[x] == prediksi[x]:
30             true_pred = true_pred + 1
31             compare.append("BENAR")
32         else:
33             compare.append("SALAH")
34     row = len(filenames)
35     list_prediksi = []
36     for i in range(row):
37         list_prediksi.append([filenames[i], prediksi[i],
38         compare[i]])
39     self.tableWidget.setRowCount(row)
40     self.tableWidget.setColumnCount(3)
41     columnHeaders = ['Nama File', 'Hasil Prediksi',
42     'Benar/Salah']
43     self.tableWidget.setHorizontalHeaderLabels(columnHeaders)
44     self.tableWidget.horizontalHeader().setSectionResizeMode(2,
45     QtWidgets.QHeaderView.Stretch)
46     for i in range(row):
47         self.addRow(i, list_prediksi[i])

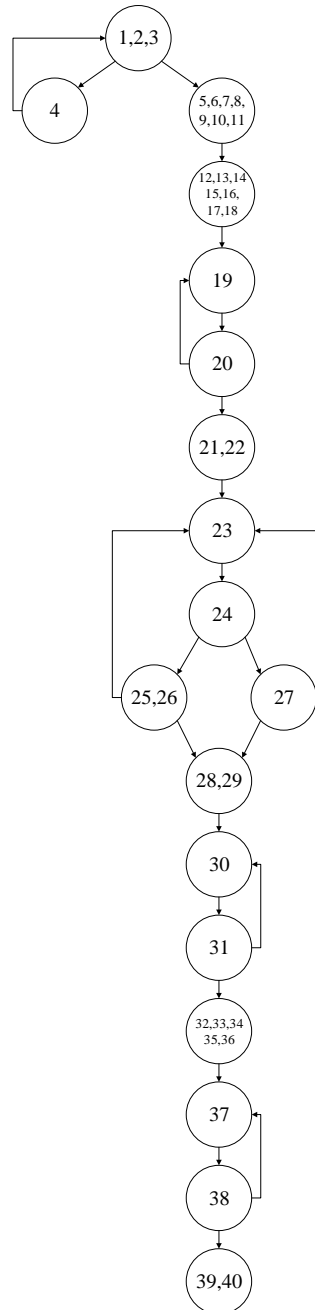
```


39	<code>persentase = (true_pred / len(filenamees)) * 100</code>
40	<code>self.lblHasil.setText("%.2f%%" % (persentase))</code>

Gambar 4.9 Source Code Proses Pengujian

a. Flow Graph

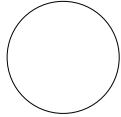

Berikut ini merupakan *Flow Graph* berdasarkan source code pada tahap sebelumnya yang digambarkan pada Gambar 4.10



Gambar 4.10 Flow Graph Proses Pengujian

Keterangan pada Gambar 4.10 dijelaskan pada Tabel 4.5 sebagai berikut.

Tabel 4.5 Keterangan *Flow Graph*

Simbol	Keterangan
	<i>Node (N)</i>
	<i>Edge (E)</i>

Berdasarkan *Flow Graph* pada Gambar 4.10 maka perhitungan *cyclometric complexity* untuk proses pelatihan adalah sebagai berikut:

$$V(G) = E - N + 2$$

$$= 23 - 17 + 2 = 8$$

Jadi, jumlah *cyclometric complexity* pada proses pelatihan adalah 8. Berdasarkan hasil tersebut, maka terdapat 8 *Independent Path* yang terdiri dari:

Path 1 = 1,2,3,4,1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,19,20,21,22,
23,24,25,26,23,24,25,26,28,29,30,31,30,31,32,33,34,35,36,37,38,37,
38,39,40

Path 2 = 1,2,3,4,1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,19,20,21,22,
23,24,27,23,24,27,28,29,30,31,30,31,32,33,34,35,36,37,38,37,38,39,40

Path 3 = 1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,19,20,21,22,23,24,
25,26,23,24,25,26,28,29,30,31,30,31,32,33,34,35,36,37,38,37,38,39,40

Path 4 = 1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,19,20,21,22,23,24,
27,23,24,27,28,29,30,31,30,31,32,33,34,35,36,37,38,37,38,39,40

Path 5 = 1,2,3,4,1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,
20,21,22,23,24,25,26,28,29,30,31,32,33,34,35,36,37,38,39,40

Path 6 = 1,2,3,4,1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,
20,21,22,23,24,27,28,29,30,31,32,33,34,35,36,37,38,39,40

Path 7 = 1,2,3,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,
21,22,23,24,25,26,28,29,30,31,32,33,34,35,36,37,38,39,40

3				1														0
4					1													0
5						1												0
6					1		1											1
7								1										0
8									1									0
9										1	1							1
10							1					1						1
11							1					1						1
12													1					0
13														1				0
14												1			1			1
15																1		0
16																	1	0
17															1		1	1
18																		
Jumlah + 1																		7

4.2.2 Pengujian *Black Box*

Pengujian fungsionalitas dilakukan dengan menggunakan metode pengujian *black box*. Pengujian *black box* memungkinkan pembangunan perangkat lunak mendapatkan sejumlah kondisi data masukan yang sepenuhnya untuk suatu program. Pengujian fungsionalitas terdiri dari rencana pengujian serta kasus dan hasil pengujian.

4.2.2.1 Rencana Pengujian

Adapun rencana pengujian *black box* yang akan dilakukan dapat dilihat pada Tabel 4.7.

Tabel 4.7 Rencana Pengujian *Black Box*

No.	Item Uji	Butir Uji	Jenis Pengujian
1	Preprocessing	Memasukan <i>file</i> citra sandi rumput	<i>Black Box</i>
		Melakukan proses pengolahan citra	<i>Black Box</i>
2	Pelatihan	Melakukan Proses Pelatihan	<i>Black Box</i>
3	Pengujian	Melakukan Proses Pengujian	<i>Black Box</i>

4.2.2.2 Kasus dan Hasil Pengujian

Tahap pengujian dilakukan dengan cara menguji setiap fungsi yang ada pada perangkat lunak untuk semua kemungkinan yang terjadi berdasarkan rencana pengujian. Berikut merupakan poin-poin pengujian yang dilakukan.

1. Pengujian memasukkan file citra sandi rumput

Berikut merupakan pengujian untuk memasukkan file citra sandi rumput yang ditunjukkan pada Tabel 4.8

Tabel 4.8 Pengujian Memasukkan File Citra Sandi Rumput

Kasus dan Hasil Pengujian (Data Benar)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Citra	A1.jpg	Citra tampil didalam aplikasi	Citra dapat ditampilkan didalam aplikasi	[√] Diterima [] Ditolak
Kasus dan Hasil Pengujian (Data Salah)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Dokumen	Document1.docx	Dokumen tidak tampil di dalam aplikasi	Dokumen tidak dapat ditampilkan di dalam aplikasi	[√] Diterima [] Ditolak

2. Pengujian proses pengolahan citra

Berikut merupakan pengujian proses pengolahan citra yang ditunjukkan pada Tabel 4.9

Tabel 4.9 Pengujian Proses Pengolahan Citra

Kasus dan Hasil Pengujian (Data Benar)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Citra, Nama Kelas	Citra : A1.jpg Nama Kelas : A	Citra diproses hingga tahap segmentasi	Citra dapat diproses hingga	[√] Diterima [] Ditolak

			tahap segmentasi	
Kasus dan Hasil Pengujian (Data Salah)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Citra, Nama Kelas	Citra : <i>Null</i> Nama Kelas : <i>Null</i>	Citra tidak diproses	Citra tidak dapat diproses	[√] Diterima [] Ditolak

3. Pengujian proses pelatihan

Berikut merupakan pengujian proses pelatihan yang ditunjukkan pada Tabel 4.10

Tabel 4.10 Pengujian Proses Pelatihan

Kasus dan Hasil Pengujian (Data Benar)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Kumpulan kelas citra, Learning Rate, Epoch	Kumpulan Kelas Citra Learning Rate : 0.0001 Epoch : 75	Kumpulan kelas citra dilatih “Pelatihan Selesai!”	Kumpulan kelas citra dapat dilatih	[√] Diterima [] Ditolak
Kasus dan Hasil Pengujian (Data Salah)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Kumpulan kelas citra, Learning Rate, Epoch	Kumpulan Kelas Citra : <i>Null</i> Learning Rate : <i>Null</i> Epoch : <i>Null</i>	Kumpulan kelas citra tidak dilatih	Kumpulan kelas citra tidak dapat dilatih	[√] Diterima [] Ditolak

4. Pengujian proses pengujian

Berikut merupakan pengujian proses pengujian yang ditunjukkan pada Tabel 4.11

Tabel 4.11 Pengujian Proses Pelatihan

Kasus dan Hasil Pengujian (Data Benar)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Kumpulan citra uji yang sudah tersegmentasi	Kumpulan Citra Uji	Kumpulan citra terklasifikasi	Kumpulan citra dapat terklasifikasi	[√] Diterima [] Ditolak
Kasus dan Hasil Pengujian (Data Salah)				
Data Uji	Data Masukan	Harapan	Keluaran	Kesimpulan
Kumpulan citra uji yang sudah tersegmentasi	Tidak Ada Kumpulan Citra Uji	Proses klasifikasi tidak berjalan	Proses klasifikasi tidak dapat berjalan	[√] Diterima [] Ditolak

4.2.2.3 Kesimpulan Pengujian Fungsionalitas Sistem

Berdasarkan hasil pengujian fungsionalitas yang telah dilakukan menggunakan metode *Black Box*, dapat ditarik kesimpulan bahwa perangkat lunak yang dibangun sudah memiliki fungsionalitas yang baik dan sudah menampilkan output yang diharapkan namun masih memungkinkan terjadi kesalahan.

4.2.3 Pengujian Akurasi

Pada tahap pengujian akurasi akan dilakukan perhitungan nilai akurasi atau kecocokan dari data baru yang masuk ke dalam aplikasi dengan data yang sudah dilatih sebelumnya. Data baru ini merupakan data yang tidak termasuk ke dalam kumpulan data latih. Pengujian dilakukan setelah proses pelatihan data training dilakukan.

Adapun metode perhitungan akurasi yang akan dilakukan yaitu dengan metode *Confusion Matrix*. *Confusion Matrix* merupakan suatu matriks yang berisi data aktual dan data prediksi yang diklasifikasikan oleh sistem. Adapun ketentuannya dapat dilihat pada Tabel 4.12

Tabel 4.12 Confusion Matrix

<i>True Label</i>	<i>Predicted Label</i>	
	0	1
0	TP	FN
1	FP	TN

Untuk menghitung akurasi terdapat 3 rumus, yaitu Akurasi, *Precision*, dan *Recall*. Adapun rumus menghitung akurasi, *presicion*, dan *recall* pada *confusion matrix* menggunakan persamaan sebagai berikut:

$$\text{Akurasi} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \times 100\%$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\%$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100\%$$

Keterangan:

0 – 1 : Index Kelas

TP : *True Positive*

TN : *True Negative*

FP : *False Positive*

FN : *False Negative*

Dikarenakan jumlah kelas yang diklasifikasikan lebih dari dua kelas, maka rumus untuk menghitung akurasi, *presicion*, dan *recall* adalah sebagai berikut [36]:

$$\text{Akurasi} = \frac{\sum_{i=1}^L \frac{\text{TP}_i + \text{TN}_i}{\text{TP}_i + \text{FN}_i + \text{FP}_i + \text{TN}_i}}{L} \times 100\%$$

$$\text{Presicion} = \frac{\sum_{i=1}^L \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i}}{L} \times 100\%$$

$$\text{Recall} = \frac{\sum_{i=1}^L \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i}}{L} \times 100\%$$

Keterangan:

TP : *True Positive* ke-i

TN : *True Negative* ke-i

FP : *False Positive* ke-i

FN : *False Negative* ke-i

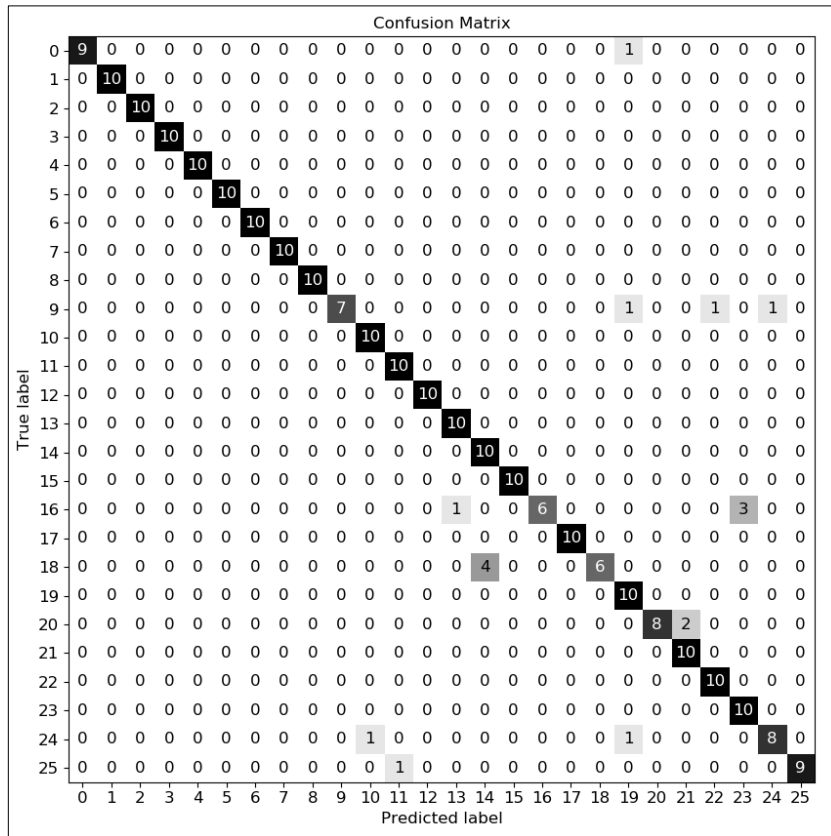
L : Jumlah Keseluruhan Data Yang Diuji/Jumlah Kelas

4.2.3.1 Proses Pengujian Akurasi

Proses pengujian dilakukan terhadap data baru yang akan dikenali oleh sistem. Sebelum ke proses pengenalan, terlebih dahulu dilakukan proses pelatihan terhadap data latih yang sudah ada. Data latih tersebut berjumlah 2600 data yang terbagi ke dalam 26 kelas yaitu kelas A sampai Z. Adapun data uji yang digunakan berjumlah 260 data. Parameter-parameter yang digunakan pada pengujian ini yaitu nilai maksimum *epoch* dan nilai *learning rate*. Untuk jumlah maksimum *epoch* ditetapkan sebesar 100 dengan nilai *learning rate* yang berubah-ubah. Berikut merupakan proses pengujian yang dilakukan.

1. Pengujian Akurasi Pertama

Pada pengujian pertama dilakukan pelatihan data latih sebanyak 2600 data dengan nilai maksimum *epoch* sebesar 100 dan nilai *learning rate* sebesar 0.0001. Untuk pengujian menggunakan data uji sebanyak 260 data. Adapun *confusion matrix* yang dihasilkan pada pengujian pengujian pertama ini dapat dilihat pada Gambar 4.12



Gambar 4.12 Confusion Matrix Pengujian Pertama

Dari *confusion matrix* diatas, dapat diukur tingkat akurasi dari prediksi dengan menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Akurasi} &= \frac{\sum_{i=1}^L \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{L} \times 100\% \\
 &= \frac{9+10+10+10+10+10+10+10+10+7+10+10+10+10+10+10+6+10+6+10+8+10+10+10+8+9}{260} \times 100\% \\
 &= \frac{243}{260} \times 100\% \\
 &= 93.46\%
 \end{aligned}$$

Untuk perhitungan *presicion* menggunakan persamaan sebagai berikut.

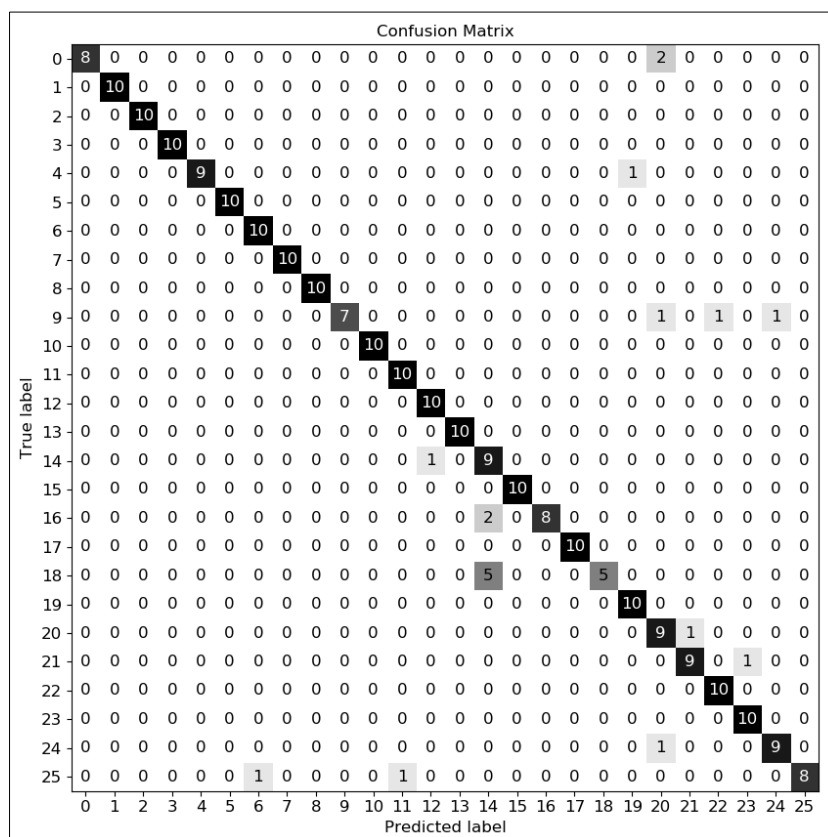
$$\begin{aligned}
 \text{Presicion} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FP_i}}{L} \times 100\% \\
 &= \frac{1+1+1+1+1+1+1+1+1+0.91+0.91+1+0.91+0.71+1+1+1+1+0.77+1+0.83+0.91+0.77+0.89+1}{26} \times 100\% \\
 &= \frac{24.61}{26} \times 100\% \\
 &= 94.66\%
 \end{aligned}$$

Sedangkan untuk perhitungan *recall* menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 Recall &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FN_i}}{L} \times 100\% \\
 &= \frac{0.9+1+1+1+1+1+1+1+1+0.7+1+1+1+1+1+0.6+1+0.6+1+0.8+1+1+1+0.8+0.9}{26} \times 100\% \\
 &= \frac{24.3}{26} \times 100\% \\
 &= 93.46\%
 \end{aligned}$$

2. Pengujian Akurasi Kedua

Pada pengujian kedua dilakukan pelatihan data latih sebanyak 2600 data dengan nilai maksimum *epoch* sebesar 100 dan nilai *learning rate* sebesar 0.0003. Untuk pengujian menggunakan data uji sebanyak 260 data. Adapun *confusion matrix* yang dihasilkan pada pengujian pengujian kedua ini dapat dilihat pada Gambar 4.13



Gambar 4.13 Confusion Matrix Pengujian Kedua

Dari *confusion matrix* diatas, dapat diukur tingkat akurasi dari prediksi dengan menggunakan persamaan sebagai berikut.

$$\begin{aligned}
\text{Akurasi} &= \frac{\sum_{i=1}^L \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{L} \times 100\% \\
&= \frac{8+10+10+10+9+10+10+10+10+7+10+10+10+10+9+10+8+10+5+10+9+9+10+10+9+8}{260} \times 100\% \\
&= \frac{241}{260} \times 100\% \\
&= 92.69\%
\end{aligned}$$

Untuk perhitungan *presicion* menggunakan persamaan sebagai berikut.

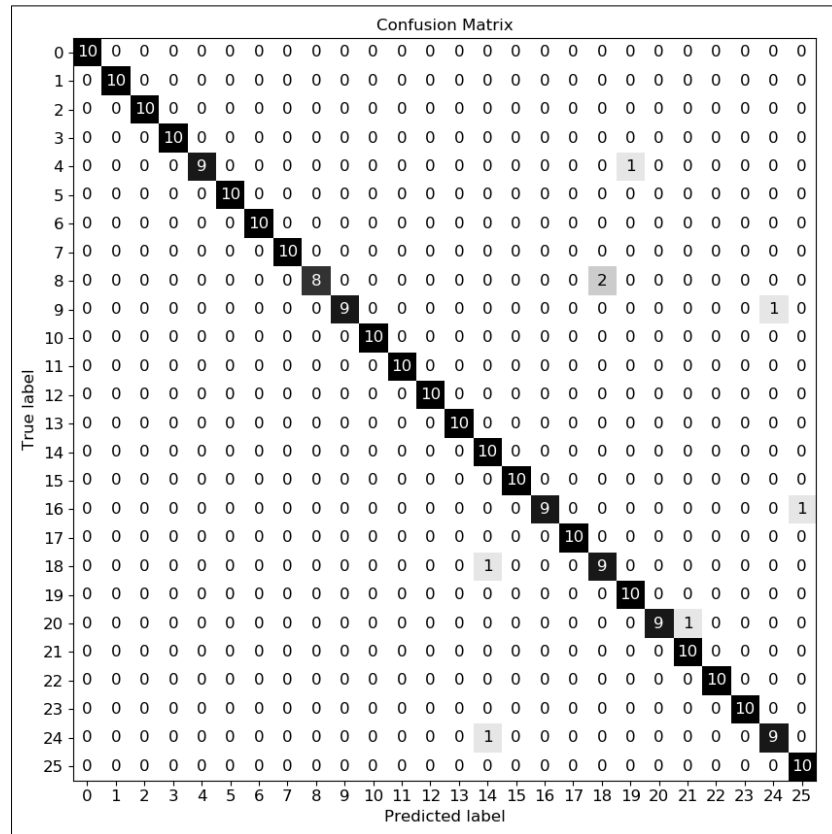
$$\begin{aligned}
\text{Presicion} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FP_i}}{L} \times 100\% \\
&= \frac{1+1+1+1+1+0.91+1+1+1+1+0.91+0.91+1+0.56+1+1+1+1+0.91+0.69+0.9+0.91+0.91+0.9+1}{26} \times 100\% \\
&= \frac{24.51}{26} \times 100\% \\
&= 94.27\%
\end{aligned}$$

Sedangkan untuk perhitungan *recall* menggunakan persamaan sebagai berikut.

$$\begin{aligned}
\text{Recall} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FN_i}}{L} \times 100\% \\
&= \frac{0.8+1+1+1+0.9+1+1+1+1+0.7+1+1+1+1+0.9+1+0.8+1+0.5+1+0.9+0.9+1+1+0.9+0.8}{26} \times 100\% \\
&= \frac{24.1}{26} \times 100\% \\
&= 92.69\%
\end{aligned}$$

3. Pengujian Akurasi Ketiga

Pada pengujian ketiga dilakukan pelatihan data latih sebanyak 2600 data dengan nilai maksimum *epoch* sebesar 100 dan nilai *learning rate* sebesar 0.0005. Untuk pengujian menggunakan data uji sebanyak 260 data. Adapun *confusion matrix* yang dihasilkan pada pengujian pengujian kedua ini dapat dilihat pada Gambar 4.14



Gambar 4.14 Confusion Matrix Pengujian Ketiga

Dari confusion matrix diatas, dapat diukur tingkat akurasi dari prediksi dengan menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Akurasi} &= \frac{\sum_{i=1}^L \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{L} \times 100\% \\
 &= \frac{10+10+10+10+9+10+10+10+8+9+10+10+10+10+10+10+9+10+9+10+9+10+10+10+9+10}{260} \times 100\% \\
 &= \frac{252}{260} \times 100\% \\
 &= 96.92\%
 \end{aligned}$$

Untuk perhitungan precision menggunakan persamaan sebagai berikut.

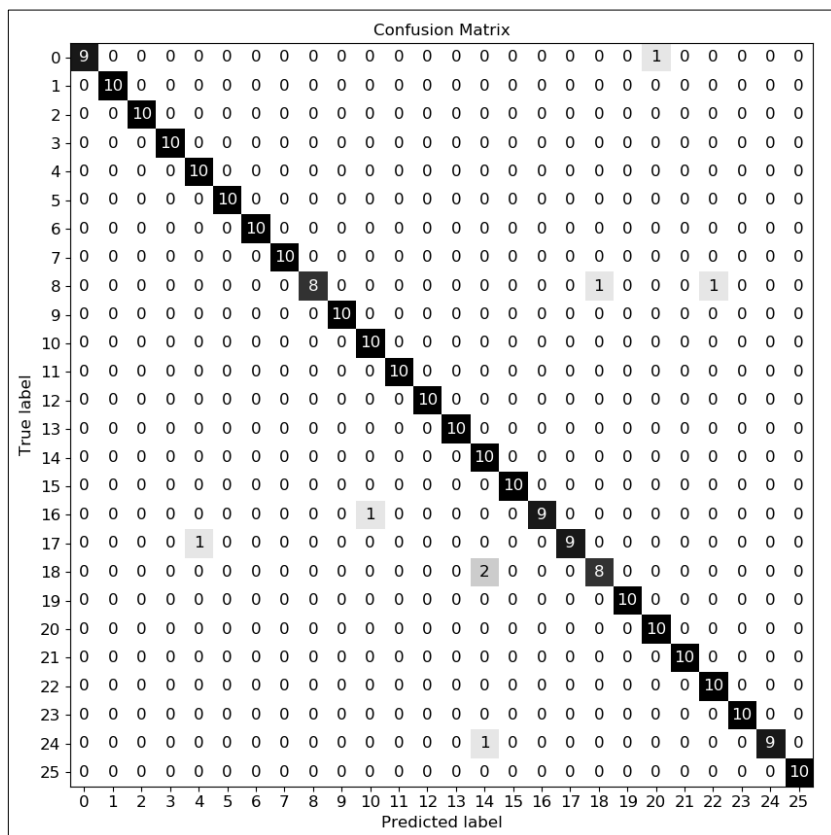
$$\begin{aligned}
 \text{Precision} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FP_i}}{L} \times 100\% \\
 &= \frac{1+1+1+1+1+1+1+1+1+1+0.83+1+1+1+0.82+0.91+1+0.91+1+1+0.9+0.91}{26} \times 100\% \\
 &= \frac{25.28}{26} \times 100\% \\
 &= 97.23\%
 \end{aligned}$$

Sedangkan untuk perhitungan recall menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Recall} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FN_i}}{L} \times 100\% \\
 &= \frac{1+1+1+1+0.9+1+1+1+0.8+0.9+1+1+1+1+1+1+0.9+1+0.9+1+0.9+1+1+1+0.9+1}{26} \times 100\% \\
 &= \frac{25.2}{26} \times 100\% \\
 &= 96.92\%
 \end{aligned}$$

4. Pengujian Akurasi Keempat

Pada pengujian keempat dilakukan pelatihan data latih sebanyak 2600 data dengan nilai maksimum *epoch* sebesar 100 dan nilai *learning rate* sebesar 0.0007. Untuk pengujian menggunakan data uji sebanyak 260 data. Adapun *confusion matrix* yang dihasilkan pada pengujian pengujian kedua ini dapat dilihat pada Gambar 4.15



Gambar 4.15 *Confusion Matrix* Pengujian Keempat

Dari *confusion matrix* diatas, dapat diukur tingkat akurasi dari prediksi dengan menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Akurasi} &= \frac{\sum_{i=1}^L \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{L} \times 100\% \\
 &= \frac{9+10+10+10+10+10+10+10+8+10+10+10+10+10+10+9+9+8+10+10+10+10+9+10}{260} \times 100\% \\
 &= \frac{252}{260} \times 100\% \\
 &= 96.92\%
 \end{aligned}$$

Untuk perhitungan *presicion* menggunakan persamaan sebagai berikut.

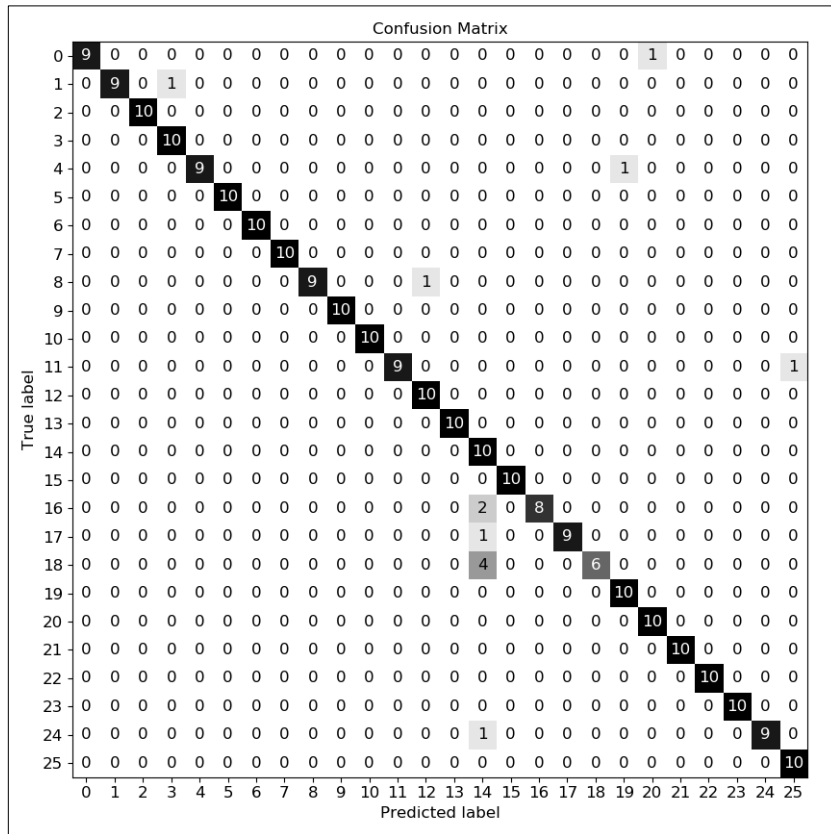
$$\begin{aligned}
 \text{Presicion} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FP_i}}{L} \times 100\% \\
 &= \frac{1+1+1+0.91+1+1+1+1+0.91+1+1+1+0.77+1+1+0.89+1+0.91+1+0.91+1+1+1}{26} \times 100\% \\
 &= \frac{25.29}{26} \times 100\% \\
 &= 97.29\%
 \end{aligned}$$

Sedangkan untuk perhitungan *recall* menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Recall} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FN_i}}{L} \times 100\% \\
 &= \frac{0.9+1+1+1+1+1+1+0.8+1+1+1+1+1+1+0.9+0.9+0.8+1+1+1+1+0.9+1}{26} \times 100\% \\
 &= \frac{25.2}{26} \times 100\% \\
 &= 96.92\%
 \end{aligned}$$

5. Pengujian Akurasi Kelima

Pada pengujian kedua dilakukan pelatihan data latih sebanyak 2600 data dengan nilai maksimum *epoch* sebesar 100 dan nilai *learning rate* sebesar 0.0009. Untuk pengujian menggunakan data uji sebanyak 260 data. Adapun *confusion matrix* yang dihasilkan pada pengujian pengujian kedua ini dapat dilihat pada Gambar 4.16



Gambar 4.16 Confusion Matrix Pengujian Kelima

Dari *confusion matrix* diatas, dapat diukur tingkat akurasi dari prediksi dengan menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Akurasi} &= \frac{\sum_{i=1}^L \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{L} \times 100\% \\
 &= \frac{9+9+10+10+9+10+10+10+9+10+10+9+10+10+10+10+8+9+6+10+10+10+10+9+10}{260} \times 100\% \\
 &= \frac{247}{260} \times 100\% = 95.00\%
 \end{aligned}$$

Untuk perhitungan *presicion* menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Presicion} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FP_i}}{L} \times 100\% \\
 &= \frac{1+1+1+0.91+1+1+1+1+1+1+1+1+0.91+1+0.56+1+1+1+1+0.91+0.91+1+1+1+1+0.91}{26} \times 100\% \\
 &= \frac{25.10}{26} \times 100\% \\
 &= 96.54\%
 \end{aligned}$$

Sedangkan untuk perhitungan *recall* menggunakan persamaan sebagai berikut.

$$\begin{aligned}
 \text{Recall} &= \frac{\sum_{i=1}^L \frac{TP_i}{TP_i + FN_i}}{L} \times 100\% \\
 &= \frac{0.9+0.9+1+1+0.9+1+1+1+0.9+1+1+1+1+0.8+0.9+0.6+1+1+1+1+0.9+1}{26} \times 100\% \\
 &= \frac{24.7}{26} \times 100\% \\
 &= 95.00\%
 \end{aligned}$$

4.2.3.2 Rekapitulasi dan Temuan Pengujian

Dari keseluruhan pengujian yang telah dilakukan, berikut merupakan hasil rekapitulasi pengujian yang ditujukan pada Tabel 4.13

Tabel 4.13 Rekapitulasi Pengujian

No	<i>Learning Rate</i>	Akurasi	<i>Precision</i>	<i>Recall</i>
1	0.0001	93.46%	94.66%	93.46%
2	0.0003	92.69%	94.27%	92.69%
3	0.0005	96.92%	97.23%	96.92%
4	0.0007	96.92%	97.29%	96.92%
5	0.0009	95.00%	96.54%	95.00%

Pada saat proses pengujian, dilakukan dua kali percobaan yang bertujuan untuk membandingkan nilai akurasi dengan konsentrasi penambahan jumlah lapisan pada arsitektur CNN tanpa merubah nilai parameter yang sudah ditentukan sebelumnya seperti nilai *epoch* dan *learning rate*. Pada percobaan pertama, *convolution layer* pada arsitektur CNN berjumlah 2 *layer* dan mendapatkan rata-rata akurasi sebesar 83%. Pada percobaan kedua, lapisan pada *convolution layer* ditambahkan 2 *layer* sehingga jumlah *layer*-nya menjadi 4 *layer*. Hasilnya didapat akurasi rata-rata sebesar 95% dan mendapat akurasi tertinggi sebesar 96.92%. Dari penjelasan tersebut ditemukan bahwa jumlah *layer* pada arsitektur CNN dapat mempengaruhi tingkat akurasi pada pengujian.

4.2.3.3 Evaluasi Hasil Pengujian Akurasi

Dari hasil pengujian terhadap 260 data uji yang telah dilakukan, evaluasi hasil akurasi yang didapatkan terhadap pengenalan pola citra sandi rumput adalah sebagai berikut.

1. Akurasi dengan persentase tertinggi didapat pada pengujian ketiga dan keempat yaitu sebesar 96.92% dengan menggunakan nilai *learning rate* sebesar 0.0005 dan 0.0007
2. Akurasi dengan persentase terendah didapat pada pengujian kedua yaitu sebesar 92.69% dengan menggunakan nilai *learning rate* sebesar 0.0003
3. Terdapat kesamaan persentase akurasi pada pengujian ketiga dan keempat yaitu sebesar 96.92% dengan menggunakan *learning rate* masing-masing 0.0005 dan 0.0007. Namun keduanya mempunyai persentase *Presicion* yang berbeda yaitu masing-masing 97.23% dan 97.29%

Berdasarkan hasil diatas, nilai *learning rate* pada proses pelatihan berpengaruh terhadap tingkat akurasi yang didapat. Persentase akurasi cenderung fluktuatif seiring semakin besarnya nilai *learning rate*. Namun, semakin kecil nilai *learning rate* membuat proses pelatihan akan semakin lambat. Kemudian faktor jumlah lapisan pada arsitektur CNN dapat mempengaruhi besaran akurasi yang didapat. Selain itu, tingkat kerapihan pola juga berpengaruh terhadap akurasi pengenalan, semakin rapih pola yang dibuat semakin mudah juga sistem mengenali pola tersebut.

