CHAPTER 5
IMPLEMENTATION AND TESTING

5.1. Implementation

This chapter five describes implementation and testing where the implementation would discover about the code and the explanation. The testing would discover about trial result of algorithms and data structures

```
1  save_to = '/media/kevin/Kevin/Skripsi/haha/data'
2  target_size = (224,224)
3  batch_size = 32
4
5  train_datagen = ImageDataGenerator(
6          featurewise_center = True,
7          featurewise_std_normalization = True,
8          rotation_range = 20,
9          width_shift_range = 0.2,
10         height_shift_range = 0.2,
11         zca_whitening = True,
12         brightness_range = (0.4,0.99),
13         validation_split = 0.2,
14         rescale = 1./255,
15         shear_range = 0.2,
16         horizontal_flip = True)
```

As we saw in the list above, line 1 is the command to specify the storage location for the results. Then lines 2 and 3 are commands to resize images. Lines 5 to 16 commands for changing images such as flip, zoom, rotate, etc.

```
1 model = Sequential()
2 model.add(Conv2D(32,(3,3),activation='relu',input_shape=(150,150, 3)))
3 model.add(MaxPooling2D(pool_size=(2, 2)))
4 model.add(Dropout(0.25))
5 model.add(Conv2D(64, (3, 3), activation='relu'))
6 model.add(MaxPooling2D(pool_size=(2, 2)))
7 model.add(Dropout(0.25))
8 model.add(Flatten())
9 model.add(Dense(96, activation='relu'))
10 model.add(Dropout(0.5))
11 model.add(Dense(3, activation='softmax'))
12 model.compile(loss='categorical_crossentropy',optimizer='Adam',metric
   s=['accuracy'])
```

As we saw in the list above, the first line means the sequential model applied to the convolution process. In line 2 to line 12 is the process where cnn works, and there are several layers consisting of several layers such as convolution layer, pooling layer, and solid layer.

```
1 epochs=100
2 batch_size=50
3
4 history = model.fit_generator(
5     train_generator,
6     epochs=epochs,
7     validation_data=validation_generator,
8     validation_steps=total_validate/batch_size,
9     steps_per_epoch=total_train/batch_size
10 )
```

The first line specifies the number of times the dataset is trained. In line 2, it determines as many as 32 data that are processed at one time. Lines 4 to 10 are commands to run the training process

```
1 predict=model.predict_generator(test_generator,steps=np.ceil(total_tes
   t/batch_size))
2 test_df['category'] = np.argmax(predict, axis=-1)
```

The code above used to predict the final result from images that want to predict

5.2. Testing

After conducting the analysis and implementation, the researcher did some testing using Relu activation. Here are some of the accuracy obtained from several tests data training.

Epochs	Training Data (%)
10	63.53 %
25	83.49 %
50	91.42 %
75	93.85 %
100	95.30%

TABLE 5.1 *Training Data*

From the results above, experiments were carried out with other activations with 100 epochs.

Epochs 100	Training Data (%)
Elu	92.78%
Tanh	37.13%

TABLE 5.2 *Training Data Using 100 Epochs*

After doing some data training, the highest percentage obtained was 95.42% at epochs 100. Furthermore, the results of the process will be saved in a file called epochs100.hdf5. The next step is to test the test image using the epochs100.hdf5 file. The images that will be tested are 50 images consisting of halfmoon, plakat, and crowntail. Testing is also carried out using Elu and Tahn activations. Here are the results of the test.

Activation	Testing (%)	Precision (%)	Recall (%)
Relu	80 %	8,79 %	8,02 %
Elu	58 %	46,56 %	58,95 %
Tanh	52 %	47,54 %	65,81 %

TABLE 5.3 *Precision Recall*

From the results above, we can see that the highest testing results were obtained by Relu activation. Therefore, of the three types of activation above, Relu activation is the most suitable for classifying betta fish species.

The following are the results of Relu activation testing :

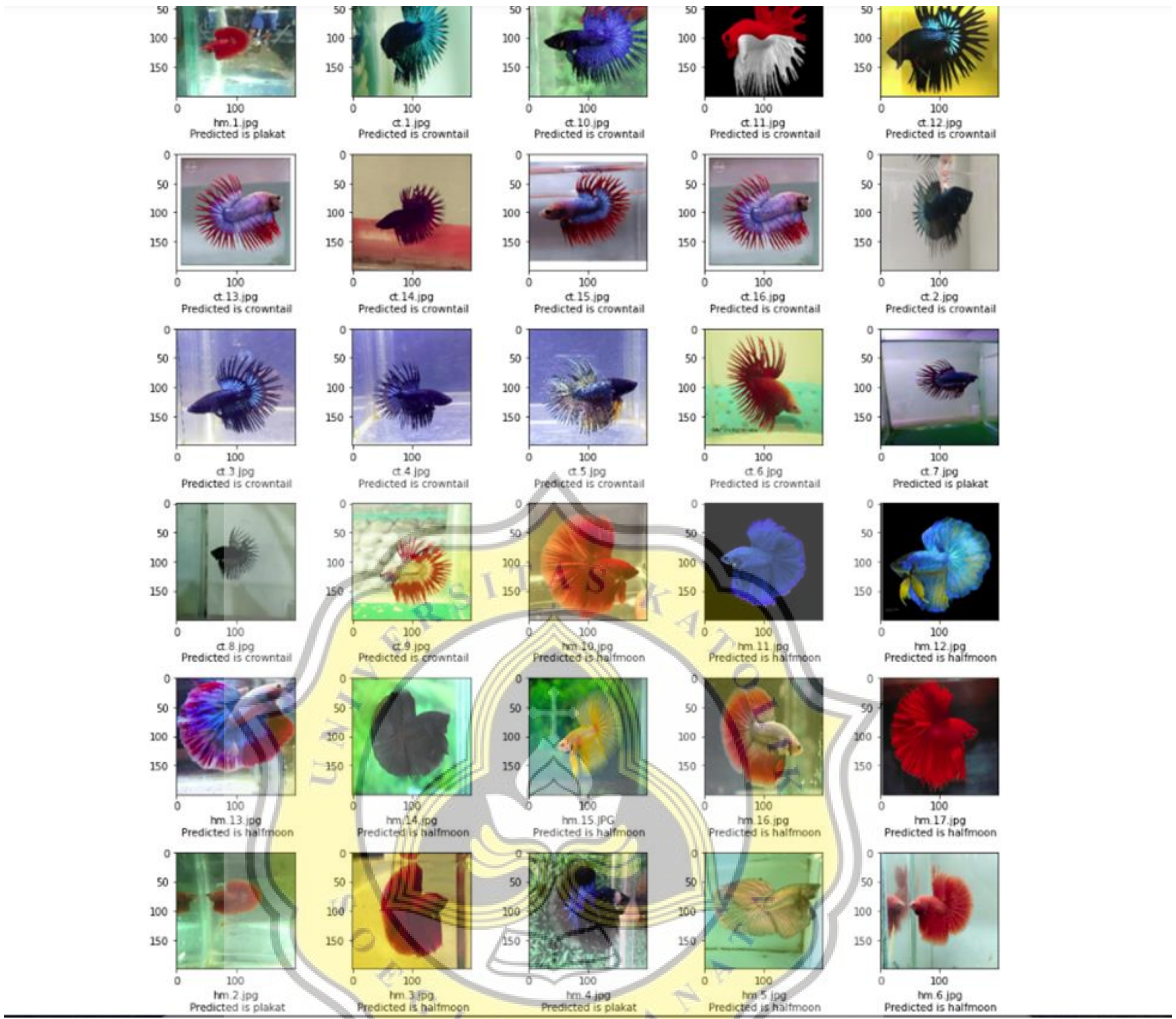
hm.1.jpg Predicted is plakat
ct.1.jpg Predicted is crowntail
ct.10.jpg Predicted is crowntail
ct.11.jpg Predicted is crowntail
ct.12.jpg Predicted is crowntail
ct.13.jpg Predicted is crowntail
ct.14.jpg Predicted is crowntail
ct.15.jpg Predicted is crowntail
ct.16.jpg Predicted is crowntail
ct.2.jpg Predicted is crowntail
ct.3.jpg Predicted is crowntail
ct.4.jpg Predicted is crowntail
ct.5.jpg Predicted is crowntail
ct.6.jpg Predicted is crowntail
ct.7.jpg Predicted is plakat
ct.8.jpg Predicted is crowntail
ct.9.jpg Predicted is crowntail
hm.10.jpg Predicted is halfmoon
hm.11.jpg Predicted is halfmoon
hm.12.jpg Predicted is halfmoon
hm.13.jpg Predicted is halfmoon
hm.14.jpg Predicted is halfmoon
hm.15.JPG Predicted is halfmoon
hm.16.jpg Predicted is halfmoon
hm.17.jpg Predicted is halfmoon
hm.2.jpg Predicted is plakat
hm.3.jpg Predicted is halfmoon
hm.4.jpg Predicted is plakat
hm.5.jpg Predicted is halfmoon
hm.6.jpg Predicted is halfmoon

FIGURE 5.1 *Test Result*

TRUE POSITIVE CT	: 15	TRUE POSITIVE PK	: 12
TRUE NEGATIVE CT	: 30	TRUE NEGATIVE PK	: 29
FALSE POSITIVE CT	: 1	FALSE POSITIVE PK	: 5
FALSE NEGATIVE CT	: 4	FALSE NEGATIVE PK	: 4
TRUE POSITIVE HM	: 13		
TRUE NEGATIVE HM	: 31		
FALSE POSITIVE HM	: 4		
FALSE NEGATIVE HM	: 2		

CROWNTAIL

precission	0.9375
recall	0.7894736842105263

HALFMOON

precission	0.7647058823529411
recall	0.8666666666666667

PLAKAT

precission	0.7058823529411765
recall	0.75

AVERAGE

ACCURACY	0.8
PRECISSION	0.802696078431373
RECALL	0.8020467836257309