Jurnal Format Volume 9 Nomor 1 Tahun 2020 :: ISSN : 2089 - 5615 :: E-ISSN : 2722 - 7162

Model Prediksi Jenis Hewan dengan Metode Convolution Neural Network

Harry Dhika[1], Nia Rahma Kurnianda[2], Puput Irfansyah[3], Wisnu Ananta[4]
Fakultas Ilmu Matematika dan Pengetahuan Alam, Program Studi Ilmu Komputer[1,2,3,4]
Fakultas Ilmu Komputer, Universitas Mercu Buana[2]
e-mail: harrydhika@apps.ipb.ac.id[1]
nia.*kurnianda@apps.ipb.ac.id[2], nia.rahma@mercubuana.ac.id[2]*
irfansyahirfansyah@apps.ipb.ac.id[3]
ananta@apps.ipb.ac.id[4]

Abstract - ***The computational process on a computer to carry out a certain task is certainly not free from learning methods. In the learning process, various methods can be carried out to be able to fulfill the training period to provide a particular computer expertise. One way to support this period is by using a deep learning convolution neural network (CNN) algorithm. CNN is able to load the entire scale of object classification information without losing its accuracy. The purpose of this study is to give computers the ability to recognize animal types and predict animal types based on the images entered. This study also aims to assess the accuracy of the learning method training outcomes compared to the learning outcomes. The method used in this study is computationally trained, a number of images of cats and dogs. Then the test will be done in the same way after going through the training convulsion stage. The results of this study the accuracy of the training results reached 97.56%***

Keywords: CNN, Deep Learning, Animal type prediction

Abstrak – Proses komputasi pada komputer untuk melaksanakan suatu tugas tertentu tentunya tidak lepas dari metode pembelajaran. Dalam proses pembelajaran, berbagai metode dapat dilakukan untuk dapat memenuhi periode training tersebut untuk memberikan komputer suatu keahlian tertentu. Salah satu cara menunjang periode tersebut adalah dengan menggunakan algoritma deep learning convolution neural network (CNN). CNN mampu memuat keseluruhan skala informasi klasifikasi objek tanpa kehilangan keakuratannya. Tujuan dari penelitian ini adalah memberikan komputer kemampuan untuk mengenali jenis binatang dan memprediksi jenis binatang berdasarkan gambar yang dimasukan. Penelitian ini juga bertujuan untuk menilai keakuratan hasil training metode pembelajaran dibangkan dengan hasil keluaran dari pembelajaran. Metode yang digunakan dalam penelitian ini adalah mentraining secara komputasi, sejumlah gambar kucing dan anjing. Kemudian test akan dilakukan dengan cara yang sama setelah melalui tahapan konvulasi training. Hasil dari penelitian ini keakuratan hasil training mencapai 97,56%

Kata Kunci : CNN, Deep Learning, Prediksi Jenis Binatang

I. PENDAHULUAN

Pada era teknologi maju saat ini, teknologi pengenalan citra menjadi salah satu inputan yang banyak diterapkan pada berbagai bidang. Hal ini dikarenakan pengenalan citra memiliki manfaat yang luas[1].

Salah satu potret fungsi dari fitur pengenalan citra adalah untuk mengenali dan memprediksi jenis obyek dalam citra. Untuk memprediksi jenis obyek pada citra, beberapa cara digunakan. Salah satunya menggunakan ANN dan CNN. Dalam penggunaan ANN tingkat akurasi hasil prediksi yang benar dalam mengenali jenis beras melalui citra beras sebesar 82%[2]. Sementara pada pengunaan CNN untuk memprediksi jenis citra notasi musik sebesar 95,56%[3]. Selain kedua metode tersebut, penelitian terdahulu membandingkan beberapa metode untuk fitur pengenalan citra dimana PCA sebesar 79%, SVM 87%, LDA 83%, LBPH 87% dan CNN 97%[4].

Berdasarkan hasil penelitian terdahulu yang telah disebutkan diatas, maka kami prediksi jenis hewan dapat dilakukan dengan menggunakan pengenalan citra diterapkan untuk melatih perangkat komputer untuk mengenali berbagai jenis binatang di muka bumi dan kami akan menerapkan metode deep learning yang memiliki rate yang tertinggi yaitu CNN.

Fitur prediksi jenis gambar tersebut nantinya akan menjadi salah satu alat penting yang sangat dibutuhkan bagi dunia penelitian karena permasalahan yang ditemukan oleh penelitian terdahulu adalah jumlah binatang yang ada di dunia ini sangat banyak, sebut saja sekitar 8,7 Juta spesies[5], dimana satu spesies ada lebih dari beberapa ribu hingga beberapa juta jenis[6].

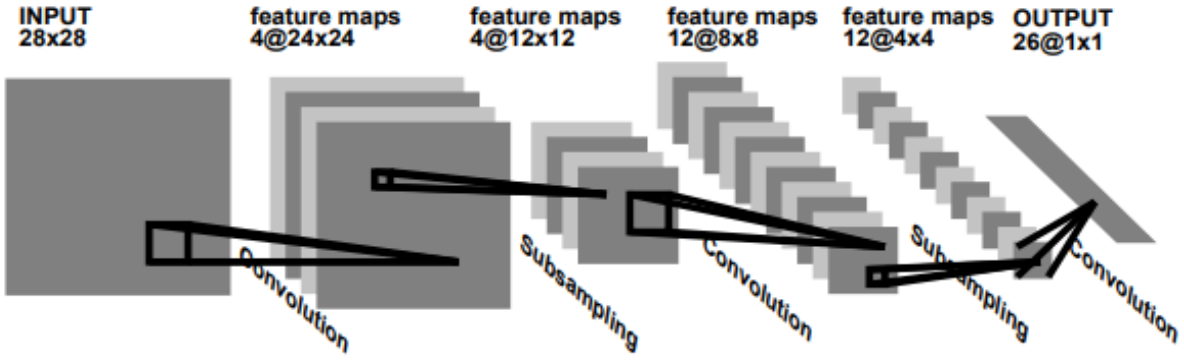
Berdasarkan permasalahan diatas, kami menemukan permasalahan lanjutan yaitu jika kita bermaksud mengenali salah satu dari spesies tersebut, kita membutuhkan banyak waktu yang jika dilakukan secara manual. Berdasarkan permasalahan tersebut, maka fitur prediksi jenis hewan yang dikembangkan ini diharapkan dapat digunakan untuk membantu peneliti dalam mengenali makhluk-mahkluk yang mereka jumpai. Fitur tersebut juga diharapkan memberikan kemudahan bagi peneliti untuk mengklasifikasikan apakah hewan yang mereka temui telah menjadi spesies yang dikenali ataupun yang tidak dikenali. Pendekatan metode yang akan digunakan untuk mengolah citra dari hewan tersebut adalah dengan menggunakan Convolution Neural Network dimana telah terbukti dalam penelitian terdahulu dapat dengan baik mengenali obyek yang ada pada gambar.

Untuk memudahkan kami dalam menjelaskan langkah-langkah penelitian kami, maka kami membagi penulisan kedalam 4 seksi. Pendahuluan, landasan teori dan metode, pekerjaan dan diskusi hasil serta kesimpulan.

II. LANDASAN TEORI DAN METODE

A. Teori CNN

Convolutional Neural Network (CNN) adalah kategori jaringan syaraf tiruan yang efektif dalam area pengenalan citra dan klasifikasi[7]. Merupakan adalah operasi khusus yang diterapkan pada matriks tertentu menggunakan matriks lain. Pada operasi matrix pengolahan citra, operasi diterapkan kepada matrix gambar (Dalam bentuk RGB)[8]. Operasi melibatkan mengalikan nilai sel yang sesuai dengan baris dan kolom tertentu, dari matriks gambar, dengan nilai sel yang sesuai dalam matriks filter. Kami melakukan ini untuk nilai-nilai semua sel dalam rentang matriks filter dan menambahkannya bersama untuk membentuk output[9].



Gambar 1. Convulation Neural Network

Dalam gambar 1 diatas, dijelaskan bahwa cara kerja Convulation Neural Network adalah dengan memberlakukan filter pada suatu gambar yang dijalankan dengan menggunakan matriks tersebut[10]. Kemudian hasil filtering tersebut difilter kembali untuk mendapatkan suatu sub sampling[11]. Kemudian sub sampling tersebut diolah ke dalam suatu pemetaan fitur. Hasil dari pemetaan tersebut diklasifikasikan ke dalam pilihan solusi yang tersedia[12].

B. Metodologi

1. **Metode Penelitian**
   Metode penelitian yang digunakan dalam studi kasus ini adalah metode terapan[13]. Penelitian ini langsung menerapkan model CNN ke dalam bahasa python sehingga dapat langsung digunakan untuk menganalisis gambar yang menjadi input.

2. **Metode Pemilihan Sampel**
   Sampel dipilih berdasarkan purposive sampling[14]. Kami menggunakan sampel binatang yang terdekat dengan jenis manusia (peliharaan) seperti anjing dan kucing. Dimana datanya dapat juga diakses secara free. Data sample yang digunakan adalah data anjing dan kucing dimana termasuk kategori binatang dengan dimensi 64x64 RGB dengan kuantitas 12.500 data training dan 200 data uji.

3. **Skenario Eksperimen**
   Eksperimen dilakukan dengan mengimplementasikan coding CNN dengan menggunakan bahasa pemrograman python 3.7 dengan langkah sebagai berikut:

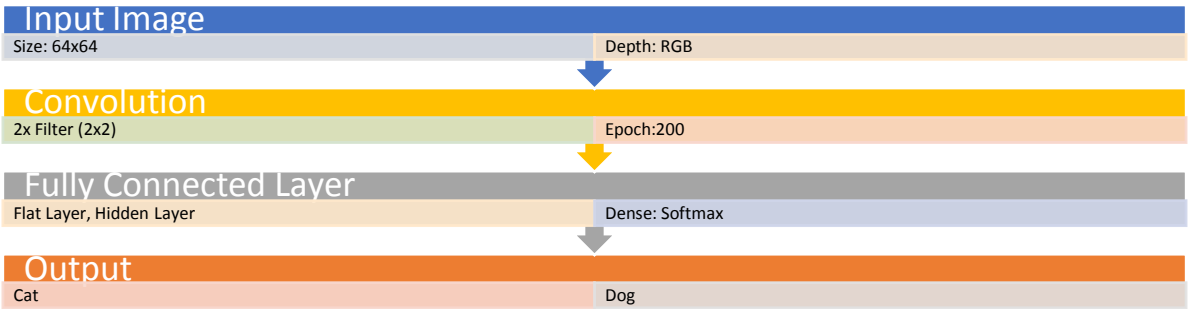
Gambar 2. Skenario Eksperimen

Pada gambar 2 diatas, skenario eksperimen dimulai dengan mendefinisikan library yang akan digunakan seperti keras dan tensorflow, kemudian data yang akan digunakan untuk mentraining dan menguji di panggil.

Langkah selanjutnya adalah dengan mentraining terlebih dahulu model menggunaakan CNN. Training diawali dengan memanggil satu persatu data gambar dan mengubahnya menjadi convolution matrix. Setelah itu barulah dilakukan pengujian dengan menggunakan 200 gambar uji untuk menilai tingkat akurasinya[15].

4. Metode Implementasi

Implementasi dilakukan dengan menggunakan framework jupiter notebook. Library dan fungsi yang digunakan antara lain menggunakan fungsi Conv2D tensorflow keras. Langkah-langkah yang dilakukan dalam implementasi antara lain adalah menerapkan langkah-langkah Convolution yang terdapat pada gambar 2. Dibawah ini:



Gambar 3. Langkah Implementasi

5. Metode Testing

Setelah data training selesai di implementasi, maka data testing berupa gambar akan di panggil untuk menguji coba hasil training. Hasil akhir dari testing adalah berupa prediksi gambar sesuai dengan jenis obyek pada gambar. Hasil keluaran testing akan dibandingkan kembali dengan hasil sebenenarnya untuk mengukur seberapa besar tingkat akurasi fitur prediksi.

III. HASIL DAN PEMBAHASAN

A. Identifikasi Alat

Pada tabel 1. Dibawah ini, kami melakukan identifikasi terhadap alat-alat yang kami pergunakan untuk melakukan implementasi terhadap arsitektur CNN yang kami buat.

Tabel 1. Identifikasi Alat

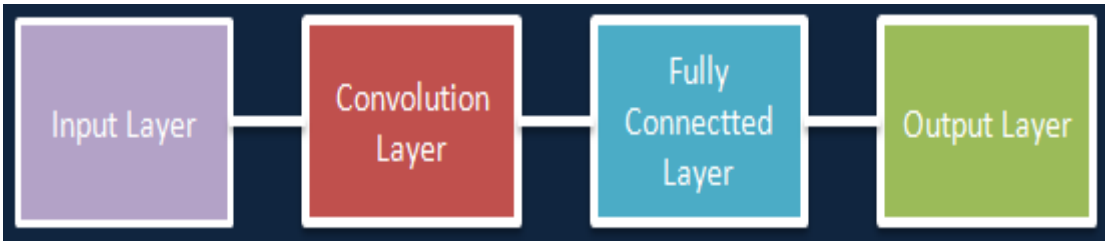
Alat	Deskripsi
Bahasa Pemrograman	Python 3.7
Aplikasi	Jupiter Notebook
Library	Matplotlib, Seaborn, Scikit-Learn, Pandas, Numpy, tensorflow keras
Data	Dog VS Cat (kaggle)
Epoch	200

B. Data yang digunakan

Sumber data yang digunakan adalah data yang disediakan oleh situs kaggle (http://www.kaggle.com/c/dogs-vs-cats/data). Sumber data ini sekumpulan gambar anjing dan kucing yang berdimensi 64x64 pixel. Data gambar dibagi menjadi dua, data training dan data testing

C. Rancangan Arsitektur Convolution Neural Network

1. Rancangan model arsitektur CNN Level 0

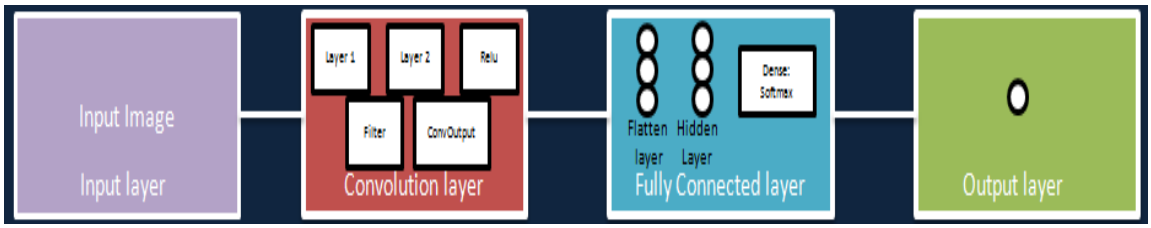


Gambar4. Model arsitektur level 0

Pada gambar 3. diatas, kami menggambarkan fungsi dasar dari model prediksi dengan fitur-fitur CNN. Dapat dilihat pada gambar tersebut, bahwa model prediksi ini pada dasarnya terdiri atas 4 layer utama, yaitu input, convolution layer, fully connected layer dan output layer

2. Rancangan model arsitektur CNN Level 1

Untuk menjelaskan komponen-komponen pendukung pada model arsitektur CNN dari model prediksi ini, kami menggunakan bantuan gambar 4 dibawah ini sebagai alat untuk menjabarkan komponen tersebut.



Gambar 5. Model arsitektur level 1

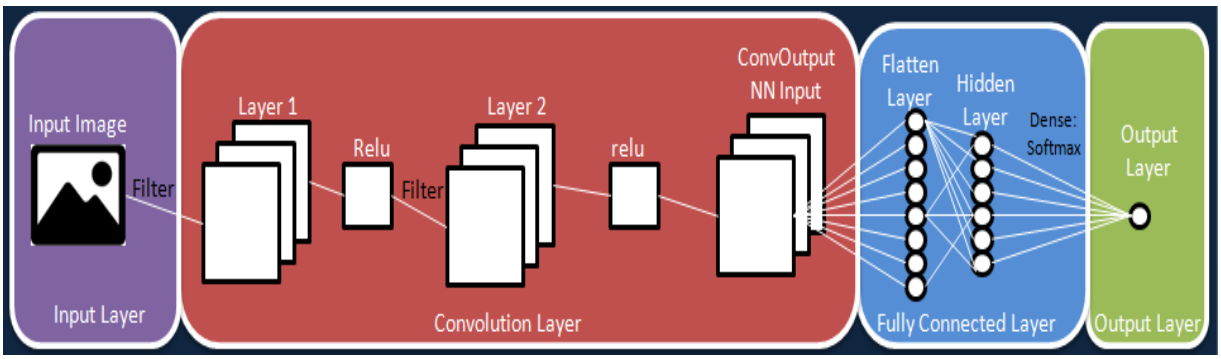
Pada gambar 4 . diatas, dapat kami jelaskan bahwa input layer terdiri atas komponen input yang berupa gambar. Sementara itu convolution layer terdiri atas layer1, layer 2, Relu, filter dan hasil convolusion. Kemudian fully connected layer memiliki komponen flatten layer dan hidden layer.

Flatten layer merupakan hasil keluaran hasil konvolusi matriks yang transformasikan ke dalam vektor[a,1] sementara hidden layer merupakan pemetaan fitur utama untuk menemukan hasil klasifikasi. Pada fully connected layer juga terdapat fungsi dense yang meningkatkan hasil prediksi pada klasifikasi kearah output yang benar.

Yang terakhir pada gambar 4 diatas adalah output layer dimana hanya terdapat 1 hasil output dari hasil keseluruhan proses convolution dan neural network tersebut.

3. Rancangan model arsitektur CNN Level 2

Untuk menjabarkan rangkaian cara kerja dari keseluruhan fungsi CNN prediksi ini, kami sajikan bentuk urutan dan koneksi yang akan dilakukan pada tahap implementasi melalui representasi turunan arsitektur level 2. Representasi tersebut kami tuangkan dalam gambar 5 dibawah ini.



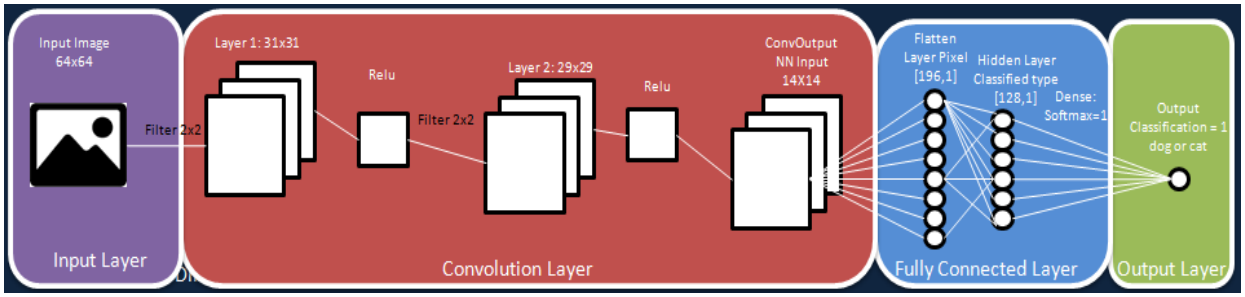
Gambar 6. Model arsitektur level 2

Pada gambar 5 diatas, kami merepresentasikan langkah-langkah kinerja CNN pada prediksi ini. Langkah kerja CNN dimulai dengan memfilter input gambar yang dimasukan, kemudian hasil filter pertama tersebut menjadi sampling di layer 1 yang kemudian dipertajam dengan fungsi relu. Kemudian hasil keluaran layer 1 difilter kembali sehingga menghasilkan sampling pada layer kedua. Hasil sampling dipertajam kembali dengan relu sehingga menjadi output proses konvulasi.

Hasil konvulasi ditransformasikan ke dalam bentuk 1 dimensi yang disebut flatten layer dan diproses oleh hidden layer yang memuat ekstraksi fitur dan memuat fungsi softmax.
Untuk kondisi lain, input dapat dikondisikan untuk menggunakan non gambar, bisa juga menggunakan teks, gambar format lain, suara dan lain sebagainya[16]

4. Rancangan model arsitektur CNN Level 3

Turunan terakhir dari model arsitektur CNN ini adalah bentuk lengkap arsitektur yang berisikan layer, komponen, rangkaian kerja dan detail rangkaian.



Gambar 7. Model arsitektur level 3

D. Klasifikasi Kelas

Tabel 2. Klasifikasi Kelas

Kelas	Karakteristik Isi
1	Kucing
2	Anjing

E. Implementasi

1. Deklarasi Library yang digunakan

```
# -*- coding: utf-8 -*-
"""
Created on Sun May 10 04:12:23 2020
@author: Harry Dhika, Puput I., Nia R
"""
import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.callbacks import TensorBoard

from warnings import filterwarnings
filterwarnings('ignore')
```

2. Proses Training

```
//network parameter dengan Relu, adam optimizer, sigmoid operation dan binary cross entropy loss
#Menggunakan CNN
classifier = Sequential()
classifier.add(Conv2D(32,(3,3),input_shape=(64,64,3),activation = 'relu'))
classifier.add(MaxPooling2D(pool_size=(2,2),strides=2)) #if stride not given it equal to pool filter size
classifier.add(Conv2D(32,(3,3),activation = 'relu'))
classifier.add(MaxPooling2D(pool_size=(2,2),strides=2))
classifier.add(Flatten())
classifier.add(Dense(units=128,activation='relu'))
classifier.add(Dense(units=1,activation='sigmoid'))
adam = tensorflow.keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0,
amsgrad=False)

classifier.compile(optimizer=adam,loss='binary_crossentropy',metrics=['accuracy'])
tensorboard = TensorBoard(log_dir="logs/{}".format(time()))

//Teknik data augmentation menggunakan shearing of images, random zoom dan horizontal flips
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rescale=1./255, shear_range=0.1,  zoom_range=0.1, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)

//Training Set
train_set = train_datagen.flow_from_directory('train',  target_size=(64,64), batch_size=32,  class_mode='binary')

//Validation Set
test_set = test_datagen.flow_from_directory('test',  target_size=(64,64), batch_size = 32, class_mode='binary',
shuffle=False)

//  uji klasifikasi jenis obyek gambar
#Test Set /no output available
test_set1 = test_datagen.flow_from_directory('test1',  target_size=(64,64), batch_size=32, shuffle=False)
```

```
Found 19998 images belonging to 2 classes.
Found 5000 images belonging to 2 classes.
Found 12500 images belonging to 1 classes.
```

Gambar 8. Hasil Uji Klasifikasi Obyek

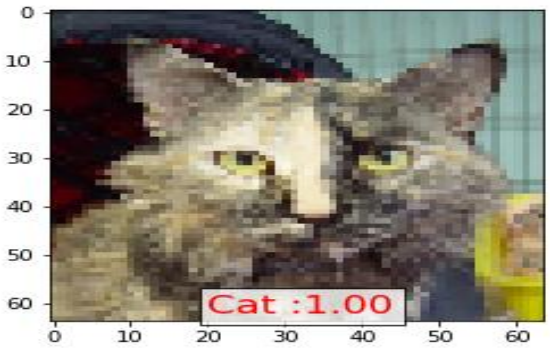
Berdasarkan gambar 7, hasil uji klasifikasinya adalah seperti yang tercantum pada tabel 3 dibawah ini:

Tabel 3. Hasil Uji Klasifikasi

Kelas	Jumlah Gambar
1	12.500
2	24.990

```
//  uji prediksi pada single image
#training set
classifier.fit_generator(train_set,  steps_per_epoch=800,  epochs  =  200,  validation_data  =  test_set,
validation_steps = 20,
                callbacks=[tensorboard]
                );

#panggil subyek tes
from tensorflow.keras.models import load_model
classifier = load_model('resources/dogcat_model_bak.h5')

#Prediksi
%matplotlib inline
import tensorflow
from tensorflow.keras.preprocessing import image
import matplotlib.pyplot as plt
import numpy as np
img1 = image.load_img('test/Cat/10.jpg', target_size=(64, 64))
img = image.img_to_array(img1)
img = img/255
# create a batch of size 1 [N,H,W,C]
img = np.expand_dims(img, axis=0)
prediction = classifier.predict(img, batch_size=None,steps=1) #gives all class prob.
if(prediction[:,:]>0.5):
   value ='Dog :%1.2f'%(prediction[0,0])
   plt.text(20, 62,value,color='red',fontsize=18,bbox=dict(facecolor='white',alpha=0.8))
else:
   value ='Cat :%1.2f'%(1.0-prediction[0,0])
   plt.text(20, 62,value,color='red',fontsize=18,bbox=dict(facecolor='white',alpha=0.8))

plt.imshow(img1)
plt.show()
```

Gambar 8 dibawah ini merupakan plot penggambaran hasil run dari uji prediksi dengan sintaks diatas jika menggunakan satuan gambar:



Gambar 9. Hasil uji prediksi satuan gambar

Berdasarkan gambar 8. Diatas, keluaran Cat:1.00 merupakan tingkat keyakinan tertinggi pada prediksi bahwa obyek pada gambar tersebut merupakan gambar kucing.

// uji prediksi pada satu batch image set (10 gambar)

```
import pandas as pd
test_set.reset
ytesthat = classifier.predict_generator(test_set)
df = pd.DataFrame({ 'filename':test_set.filenames, 'predict':ytesthat[:,0],  'y':test_set.classes
})

pd.set_option('display.float_format', lambda x: '%.5f' % x)
df['y_pred'] = df['predict']>0.5
df.y_pred = df.y_pred.astype(int)
df.head(10)
```

Keluaran dari hasil run sintaks terhadap 10 gambar diatas digambarkan pada tabel 4 dibawah ini:

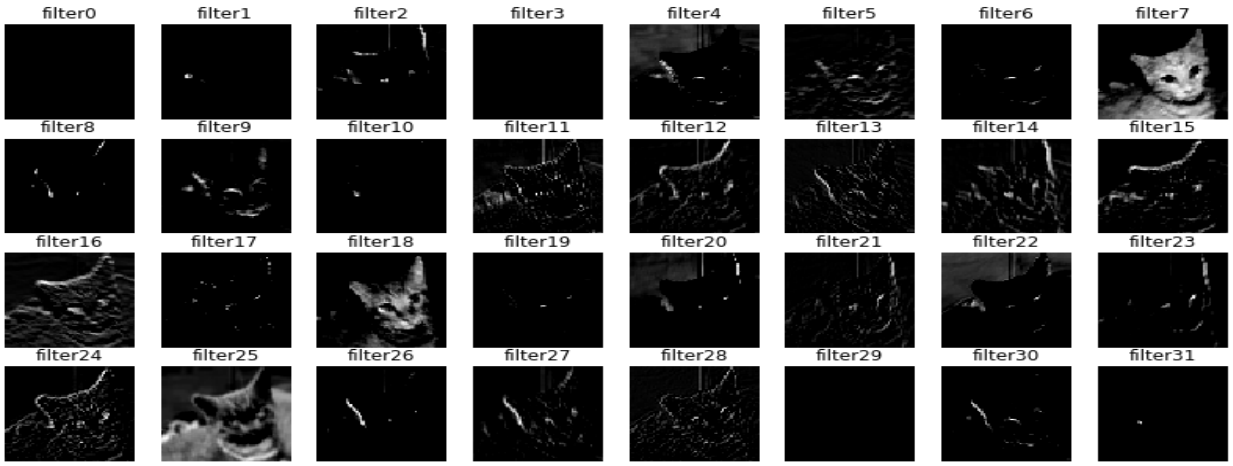
Tabel 4. Hasil Prediksi pada Set Image

Indeks	Filename	Predict	Y	Y_Predict
0	Cat/0.jpg	0.00000	0	0
1	Cat/1.jpg	0.00000	0	0
2	Cat/10.jpg	0.00000	0	0
3	Cat/100.jpg	0.99970	0	1
4	Cat/10001.jpg	0.00000	0	0
5	Cat/10009.jpg	0.02340	0	0
6	Cat/1001.jpg	0.00001	0	0
7	Cat/10012.jpg	0.00000	0	0
8	Cat/10017.jpg	0.00000	0	0
9	Cat/10018.jpg	0.00000	0	0

3. Uji Model CNN

```
#Input Image for Layer visualization
img1 = image.load_img('test/Cat/14.jpg')
plt.imshow(img1);

#preprocess image
img1 = image.load_img('test/Cat/14.jpg', target_size=(64, 64))
img = image.img_to_array(img1)
img = img/255
img = np.expand_dims(img, axis=0)
```

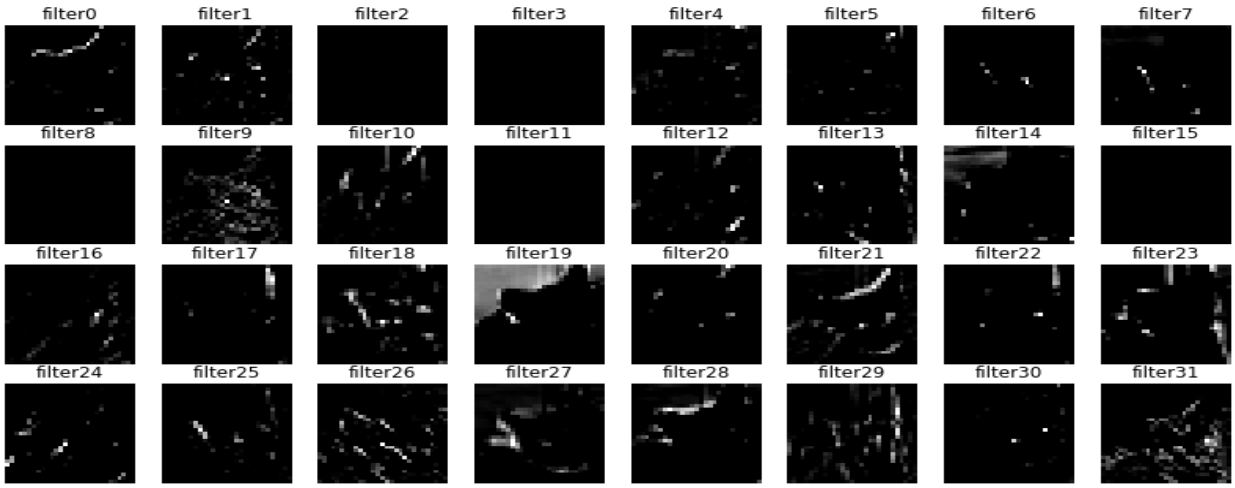
```
#memberikan model layer
model_layers = [ layer.name for layer in classifier.layers]
print('layer name : ',model_layers)

#memberi nama pada layer
layer name :  ['conv2d_6', 'max_pooling2d_6', 'conv2d_7', 'max_pooling2d_7', 'flatten_3', 'dense_6', 'dense_7']

#membuat model convolusion dengan keras
from tensorflow.keras.models import Model
conv2d_6_output = Model(inputs=classifier.input, outputs=classifier.get_layer('conv2d_6').output)
conv2d_7_output = Model(inputs=classifier.input,outputs=classifier.get_layer('conv2d_7').output)
conv2d_6_features = conv2d_6_output.predict(img)
conv2d_7_features = conv2d_7_output.predict(img)
print('First conv layer feature output shape : ',conv2d_6_features.shape)
print('First conv layer feature output shape : ',conv2d_7_features.shape)

#membuat output shape
First conv layer feature output shape :  (1, 62, 62, 32)
First conv layer feature output shape :  (1, 29, 29, 32)

#convolution pada layer 1
plt.imshow(conv2d_6_features[0, :, :, 4], cmap='gray')
<matplotlib.image.AxesImage at 0x7f3b1c90f978>
import matplotlib.image as mpimg

fig=plt.figure(figsize=(14,7))
columns = 8
rows = 4
for i in range(columns*rows):
    #img = mpimg.imread()
    fig.add_subplot(rows, columns, i+1)
    plt.axis('off')
    plt.title('filter'+str(i))
    plt.imshow(conv2d_6_features[0, :, :, i], cmap='gray')
plt.show()
```



Gambar 10. Hasil Convolution layer 1

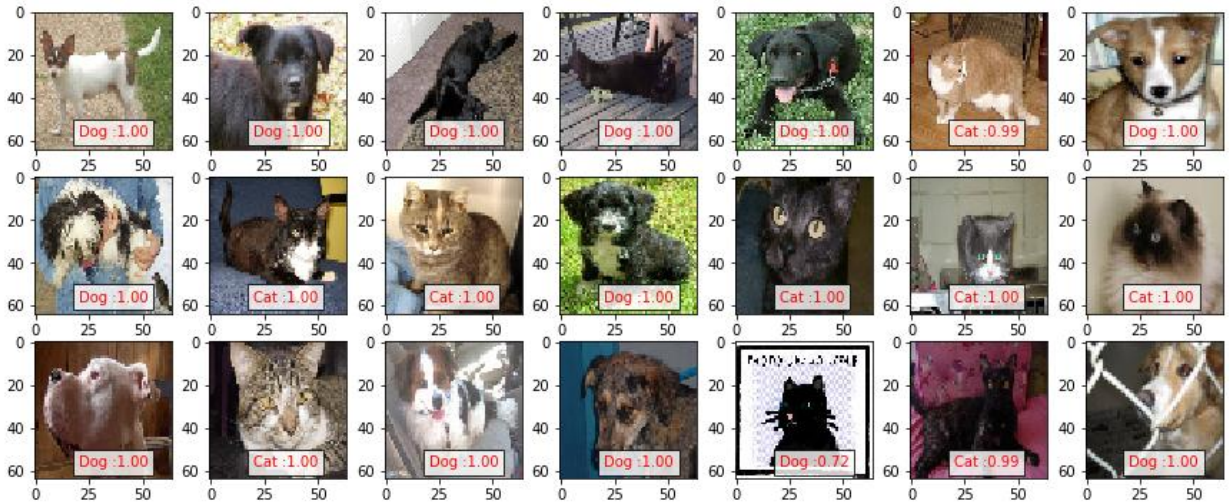
```
#Convolution layer 2
fig=plt.figure(figsize=(14,7))
columns = 8
rows = 4
for i in range(columns*rows):
    #img = mpimg.imread()
    fig.add_subplot(rows, columns, i+1)
    plt.axis('off')
```

```
    plt.title('filter'+str(i))
    plt.imshow(conv2d_7_features[0, :, :, i], cmap='gray')
plt.show()
```



Gambar 11. Hasil Convolusion Layer 2

```
#melihat model performance
fig=plt.figure(figsize=(15, 6))
columns = 7
rows = 3
for i in range(columns*rows):
    fig.add_subplot(rows, columns, i+1)
    img1 = image.load_img('test1/'+test_set1.filenames[np.random.choice(range(12500))], target_size=(64, 64))
    img = image.img_to_array(img1)
    img = img/255
    img = np.expand_dims(img, axis=0)
    prediction = classifier.predict(img, batch_size=None,steps=1) #gives all class prob.
    if(prediction[:,:]>0.5):
        value ='Dog :%1.2f'%(prediction[0,0])
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white',alpha=0.8))
    else:
        value ='Cat :%1.2f'%(1.0-prediction[0,0])
        plt.text(20, 58,value,color='red',fontsize=10,bbox=dict(facecolor='white',alpha=0.8))
    plt.imshow(img1)
```



Gambar 12. Hasil Performance Prediksi CNN

4. Uji Akurasi Model CNN

```
x1 = classifier.evaluate_generator(train_set)
x2 = classifier.evaluate_generator(test_set)
print('Training Accuracy  : %1.2f%%     Training loss  : %1.6f'%(x1[1]*100,x1[0]))
print('Validation Accuracy: %1.2f%%     Validation loss: %1.6f'%(x2[1]*100,x2[0]))
```

```
Training Accuracy  : 99.96%     Training loss  : 0.002454
Validation Accuracy: 97.56%     Validation loss: 0.102678
```

Gambar 13. Akurasi Model

IV. KESIMPULAN

Berdasarkan hasil implementasi dan ujicoba pada model diatas, maka kami mendapatkan kesimpulan bahwa model tersebut dapat digunakan untuk mentraining sejumlah besar data berupa gambar dan mampu mengenali dengan akurasi sebesar 97,56% dengan hasil akurasi yang tinggi, maka tingkat kerumitan model dapat ditambahkan dengan mengenali berbagai macam jenis binatang yang sudah terdata di dunia. Sehingga dapat membantu peneliti untuk discover jenis binatang lain yang ditemui

V. REFERENSI

[1] D. Indarti, "KARAKTERISTIK KURVA SEDERHANA OBJECT RECOGNITION IN IMAGE BASED ON SIMILARITY," vol. 20, no. 100.

[2] D. Ricardo and G. Gasim, "Perbandingan Akurasi Pengenalan Jenis Beras dengan Algoritma Propagasi Balik pada Beberapa Resolusi Kamera," *J. RESTI (Rekayasa Sist. dan Teknol. Informasi)*, vol. 3, no. 2, pp. 131–140, 2019, doi: 10.29207/resti.v3i2.894.

[3] D. M. Hakim and E. Rainarli, "Convolutional Neural Network untuk Pengenalan Citra Notasi Musik," *Techno.Com*, vol. 18, no. 3, pp. 214–226, 2019, doi: 10.33633/tc.v18i3.2387.

[4] T. Trnovszky, P. Kamencay, R. Orjesek, M. Benco, and P. Sykora, "Animal recognition system based on convolutional neural network," *Adv. Electr. Electron. Eng.*, vol. 15, no. 3, pp. 517–525, 2017, doi: 10.15598/aeee.v15i3.2202.

[5] C. Mora, D. P. Tittensor, S. Adl, A. G. B. Simpson, and B. Worm, "How many species are there on earth and in the ocean?," *PLoS Biol.*, vol. 9, no. 8, pp. 1–8, 2011, doi: 10.1371/journal.pbio.1001127.

[6] Y. H. Zhang and X. Jia, "Republication of conference papers in journals?," *Learn. Publ.*, vol. 26, no. 3, pp. 189–196, 2013, doi: 10.1087/20130307.

[7] J. L. Wu and W. Y. Ma, "A Deep Learning Framework for Coreference Resolution Based on Convolutional Neural Network," *Proc. - IEEE 11th Int. Conf. Semant. Comput. ICSC 2017*, pp. 61–64, 2017, doi: 10.1109/ICSC.2017.57.

[8] Z. Li, G. Chen, and T. Zhang, "Temporal attention networks for multitemporal multisensor crop classification," *IEEE Access*, vol. 7, pp. 134677–134690, 2019, doi: 10.1109/ACCESS.2019.2939152.

[9] Y. Lecun and Y. Bengio, "Convolutional Neural Networks for Images, Speech and Time-Series," *AT&T Bell Lab.*, 1995.

[10] T. Purwaningsih, T. Nurhikmat, and P. B. Utami, "Image classification of Golek puppet images using convolutional neural networks algorithm," *Int. J. Adv. Soft Comput. its Appl.*, vol. 11, no. 1, pp. 34–45, 2019.

[11] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

[12] A. Chatterjee, J. Saha, J. Mukherjee, S. Aikat, and A. Misra, "Unsupervised Land Cover Classification of Hybrid and Dual-Polarized Images Using Deep Convolutional Neural Network," *IEEE Geosci. Remote Sens. Lett.*, pp. 1–5, 2020, doi: 10.1109/lgrs.2020.2993095.

[13] K. Yin, Robert, *Qualitative Research from Start to Finish 2nd Edition*, 2nd ed. New York, 2016.

[14] A. Agarwal, S. Gupta, and T. Choudhury, "Continuous and Integrated Software Development using DevOps," *2018 Int. Conf. Adv. Comput. Commun. Eng.*, no. June, pp. 290–293, 2018, doi: 10.1109/icacce.2018.8458052.

[15] S. Wolfert, L. Ge, C. Verdouw, and M. J. Bogaardt, "Big Data in Smart Farming – A review," *Agric. Syst.*, vol. 153, pp. 69–80, 2017, doi: 10.1016/j.agsy.2017.01.023.

[16] W. Gunawan, "Pengembangan Aplikasi Berbasis Android Untuk Pengenalan Huruf Hijaiyah," vol. 6, no. 1, pp. 69–76, 2019.