SANTIKA
Seminar Nasional Informatika Bela Negara (SANTIKA)
ISSN (Online) 2747-0563
Volume x tahun 2024

Klasifikasi Jenis Anjing Dengan Metode Neural Network

M. Naufal Rafy Ghaly[1], Sandy Aryadika Widodo[2*], Andhika Puja Kusuma Anggara[3]

[1,2,3] Informatika, Universitas Pembangunan Nasional Veteran Jatim
[1]22081010295@student.upnjatim.ac.id
[2]22081010323@student.upnjatim.ac.id
[3]22081010335@student.upnjatim.ac.id

[2] Informatika, Universitas Pembangunan Nasional Veteran Jatim
basukirahmat.if@upnjatim.ac.id

***Abstrak*— Klasifikasi jenis anjing menimbulkan tantangan dalam pengenalan gambar karena variasi pose, ukuran, dan warna. Penelitian ini menggunakan metode Neural Network, khususnya Convolutional Neural Networks (CNN), untuk meningkatkan akurasi klasifikasi jenis anjing. Dataset yang digunakan mencakup ribuan gambar berbagai jenis anjing. Pelatihan model melibatkan augmentasi data untuk meningkatkan generalisasi dan teknik seperti regularisasi dan dropout untuk mencegah penyesuaian berlebihan. Hasil eksperimen menunjukkan bahwa model CNN mencapai akurasi tinggi, melampaui metode klasifikasi tradisional. Implementasi ini memiliki potensi aplikasi dalam pengenalan spesies anjing secara otomatis dan sistem adopsi hewan peliharaan berbasis teknologi.**

Kata Kunci— **Klasifikasi Jenis Anjing, Convolutional Neural Networks (CNN), Augmentasi Data, Regularisasi, Pengenalan Gambar.**

I. PENDAHULUAN

Pengenalan gambar adalah bidang utama dalam kecerdasan buatan dan pemrosesan gambar yang telah berkembang secara signifikan dalam beberapa tahun terakhir. Salah satu aplikasi yang menarik dari teknologi ini adalah klasifikasi ras anjing, yang memiliki banyak kegunaan praktis, seperti mengidentifikasi ras untuk adopsi hewan peliharaan, memantau hewan di taman, dan bahkan mendiagnosis masalah kesehatan pada hewan peliharaan.

Meskipun tugas klasifikasi ini mungkin tampak mudah, namun sebenarnya cukup menantang karena berbagai faktor. Anjing memiliki berbagai bentuk, ukuran, warna, dan pola bulu, sehingga menciptakan keragaman visual yang signifikan dalam foto mereka. Selain itu, faktor-faktor seperti pose, pencahayaan, dan latar belakang dapat memengaruhi akurasi klasifikasi. Metode tradisional yang mengandalkan pengenalan fitur secara manual sering kali kesulitan untuk menangani kerumitan ini secara efektif.

Untuk mengatasi masalah ini, Neural Networks, khususnya Convolutional Neural Networks (CNN), telah menjadi pilihan utama dalam pengenalan gambar. CNN dikenal dengan kemampuannya yang luar biasa untuk mengekstrak fitur-fitur penting dari gambar melalui lapisan convolutional dan pooling. Dengan pelatihan yang tepat, CNN dapat mengidentifikasi pola yang kompleks dan bervariasi dalam dataset gambar.

Penelitian ini bertujuan untuk mengeksplorasi keefektifan CNN dalam tugas klasifikasi jenis anjing. Kami menggunakan dataset yang beragam dan kaya akan variasi untuk melatih model CNN, dengan harapan model tersebut dapat menggeneralisasi dengan baik untuk gambar anjing yang tidak termasuk dalam set pelatihan. Selain itu, kami menerapkan berbagai teknik augmentasi dan regularisasi data untuk meningkatkan kinerja dan mencegah penyesuaian yang berlebihan.

Melalui penelitian ini, kami berharap dapat memberikan kontribusi yang signifikan pada bidang pengenalan gambar dan menunjukkan potensi besar jaringan saraf dalam aplikasi dunia nyata yang terkait dengan pengenalan spesies hewan. Hasil yang kami peroleh dapat membuka jalan untuk implementasi lebih lanjut dari teknologi ini dalam berbagai konteks, membawa manfaat praktis bagi masyarakat yang lebih luas.

II. METODE

Penelitian kami melibatkan penggunaan Convolutional Neural Networks (CNN) untuk mengklasifikasikan berbagai jenis anjing. Inilah cara kami melakukannya:

1. Pengumpulan Data: Kami memanfaatkan kumpulan data Kaggle dengan 10.000 gambar anjing, yang dibagi dengan rapi ke dalam set pelatihan, validasi, dan pengujian.

2. Pemrosesan Data:
   - Kami memulai dengan mengimpor pustaka seperti numpy, pandas, matplotlib.
   - Kemudian, kami mengunggah file CSV dengan label gambar.
   - Selanjutnya, kami menormalkan nilai piksel dan menggunakan pengodean satu-hot untuk label.

3. Pembangunan Model:

ISSN (Online) 2747-0563
Volume x tahun 2024

- Model CNN kami dibangun dengan beberapa lapisan konvolusi dan penyatuan untuk mengekstrak fitur dari gambar. Hal ini diikuti dengan lapisan padat untuk klasifikasi yang sebenarnya.
- Untuk membuat model yang lebih kuat, kami menggunakan teknik augmentasi data dan dropout untuk menangkis overfitting.

4. Pelatihan:
   - Kami membagi dataset menjadi set pelatihan, validasi, dan pengujian.
   - Model ini dilatih lebih dari 100 epoch dengan ukuran batch 128, dan kami mengawasi kinerja set validasi secara keseluruhan.

5. Evaluasi:
   - Akurasi model diuji dengan set data uji.
   - Kami juga memvisualisasikan riwayat pelatihan untuk memeriksa tanda-tanda overfitting.

Menggunakan CNN bersama dengan teknik augmentasi data memungkinkan kami untuk mencapai akurasi klasifikasi yang tinggi, menunjukkan kemampuan model untuk menggeneralisasi dengan baik dengan data baru.

III. HASIL DAN PEMBAHASAN

Kami menggunakan dataset yang berisi 10000 data anjing milik Catherine Horng di Kaggle

1. Mengimport library yang dibutuhkan.

Important liprary imports

```
# Importing Library
from google.colab import files
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from tqdm import tqdm
from keras.preprocessing import image
from sklearn.preprocessing import label_binarize
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
from keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import load_img
```

Gbr. 1 Import Library

Bagian ini mengimpor berbagai pustaka yang diperlukan, termasuk pustaka untuk manipulasi data (numpy, pandas), visualisasi (matplotlib), preprocessing gambar (keras.preprocessing.image, tensorflow.keras.preprocessing.image), dan pembuatan serta pelatihan model neural network (keras.models, keras.layers).

2. Mengupload file data CSV yang sudah di download dari kaggle

Loading the labels data into dataframe and viewing the data.

```
files.upload()
```

Gbr. 2 Upload Data CSV

Baris ini membuka dialog untuk mengunggah data label (labels.csv) dan menampilkannya.

3. Membaca file CSV

```
In [ ]: # Read the labels.csv file and check shape and records
        labels_all = pd.read_csv('labels.csv')
        print(labels_all.shape)
        labels_all.head()
```

```
(10222, 2)
```

Out[ ]:

	id	breed
0	000bec180eb18c7604dcecc8fe0dba07	boston_bull
1	001513dfcb2ffafc82cccf4d8bbaba97	dingo
2	001cdf01b096e06d78e9e5112d419397	pekinese
3	00214f311d5d2247d5dfe4fe24b2303d	bluetick
4	0021f9ceb3235effd7fcde7f7538ed62	golden_retriever

Gbr. 3 Code dan Output Dari Membaca File CSV

Bagian ini membaca file CSV yang berisi label gambar, menampilkan bentuk data, dan beberapa baris pertama dari dataset tersebut.

4. Menghitung distribusi ras

```
In [ ]: # Loading number or each breed
        breed_all = labels_all['breed']
        breed_count = breed_all.value_counts()
        breed_count.head()
```

```
Out[ ]: breed
        scottish_deerhound      126
        maltese_dog             117
        afghan_hound            116
        entlebucher             115
        bernese_mountain_dog    114
        Name: count, dtype: int64
```

Gbr. 4 Code dan Output Dari dan Untuk Distribusi Ras

Menghitung jumlah setiap jenis anjing dalam dataset dan menampilkan beberapa jenis anjing dengan jumlah terbanyak.

5. Memilih ras anjing secara spesifik

```
In [ ]: # Selecting all breeds because i have high computation power
        CLASS_NAME = ['scottish_deerhound', 'maltese_dog', 'afghan_hound', 'entlebucher', 'bernese_mountain_dog']
        labels = labels_all[(labels_all['breed'].isin(CLASS_NAME))]
        labels = labels.reset_index()
        labels.head()
```

Out[ ]:

	index	id	breed
0	9	0042188c895a2f14ef64a918ed9c7b64	scottish_deerhound
1	12	00693b8bc2470375cc744a6391d397ec	maltese_dog
2	79	01e787576c003930f96c966f9c3e1d44	scottish_deerhound
3	80	01ee3c7ff9bcaba98741831358776 70e	entlebucher
4	88	021b5a49189665c0442c19b5b33e8cf1	entlebucher

Gbr. 5 Code dan Output Dari dan Untuk Memilih Ras Anjing

ISSN (Online) 2747-0563
Volume x tahun 2024

Memilih beberapa jenis anjing spesifik untuk klasifikasi dan mereset indeks dataset.

6. Mengupload gambar anjing

```
In [ ]: files.upload()
```

Gbr. 6 Upload Gambar Anjing

Mengunggah gambar-gambar anjing yang akan digunakan dalam pelatihan model.

7. Membuat dan normalisasi data

```
In [ ]: # Creating numpy matrix with zeros
X_data = np.zeros((len(labels), 224, 224, 3), dtype='float32')
# One hot encoding
Y_data = label_binarize(labels['breed'], classes = CLASS_NAME)

# Reading and converting image to numpy array and normalizing dataset
for i in tqdm(range(len(labels))):
    img_path = 'shiba.jpg'  # Replace with the actual path to your image
    img = load_img(img_path, target_size=(224, 224))
    img = image.img_to_array(img)
    x = np.expand_dims(img.copy(), axis=0)
    X_data[i] = x / 255.0

# Printing train image and one hot encode shape & size
print('\nTrain Images shape: ',X_data.shape,' size: {:,}'.format(X_data.size))
print('One-hot encoded output shape: ',Y_data.shape,' size: {:,}'.format(Y_data.size))

100%|██████████| 588/588 [00:01<00:00, 444.66it/s]
Train Images shape:  (588, 224, 224, 3)  size: 88,510,464
One-hot encoded output shape:  (588, 5)  size: 2,940
```

Gbr. 7 Code dan Output Dari dan Untuk Normalisasi Data

Membuat matriks numpy untuk gambar dengan ukuran yang diinginkan, melakukan one-hot encoding pada label, memuat gambar, mengonversinya menjadi array numpy, dan menormalisasi nilai piksel.

8. Membangun model CNN

```
In [ ]: # Building the Model
model = Sequential()

model.add(Conv2D(filters = 64, kernel_size = (5,5), activation ='relu', input_shape = (224,224,3)))
model.add(MaxPool2D(pool_size=(2,2)))

model.add(Conv2D(filters = 32, kernel_size = (3,3), activation ='relu', kernel_regularizer = 'l2'))
model.add(MaxPool2D(pool_size=(2,2)))

model.add(Conv2D(filters = 16, kernel_size = (7,7), activation ='relu', kernel_regularizer = 'l2'))
model.add(MaxPool2D(pool_size=(2,2)))

model.add(Conv2D(filters = 8, kernel_size = (5,5), activation ='relu', kernel_regularizer = 'l2'))
model.add(MaxPool2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dense(128, activation = "relu", kernel_regularizer = 'l2'))
model.add(Dense(64, activation = "relu", kernel_regularizer = 'l2'))
model.add(Dense(len(CLASS_NAME), activation = "softmax"))

model.compile(loss = 'categorical_crossentropy', optimizer = Adam(0.0001),metrics=['accuracy'])

model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 220, 220, 64)      4864

 max_pooling2d (MaxPooling2  (None, 110, 110, 64)      0
 D)

 conv2d_1 (Conv2D)           (None, 108, 108, 32)      18464

 max_pooling2d_1 (MaxPoolin  (None, 54, 54, 32)        0
 g2D)

 conv2d_2 (Conv2D)           (None, 48, 48, 16)        25104

 max_pooling2d_2 (MaxPoolin  (None, 24, 24, 16)        0
 g2D)

 conv2d_3 (Conv2D)           (None, 20, 20, 8)         3208

 max_pooling2d_3 (MaxPoolin  (None, 10, 10, 8)         0
 g2D)

 flatten (Flatten)           (None, 800)               0

 dense (Dense)               (None, 128)               102528

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 5)                 325

=================================================================
Total params: 162749 (635.74 KB)
Trainable params: 162749 (635.74 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

After defining the network architecture we found out the total parameters as 162,619.

Gbr. 8 Code dan Output Dari dan Untuk Membangun CNN

Membangun arsitektur CNN dengan beberapa lapisan konvolusi dan pooling untuk mengekstraksi fitur dari gambar, diikuti dengan lapisan dense untuk klasifikasi. Model dikompilasi menggunakan fungsi loss categorical cross-entropy dan optimizer Adam.

9. Membagi dataset

```
In [ ]: # Splitting the data set into training and testing data sets
X_train_and_val, X_test, Y_train_and_val, Y_test = train_test_split(X_data, Y_data, test_size = 0.1)
# Splitting the training data set into training and validation data sets
X_train, X_val, Y_train, Y_val = train_test_split(X_train_and_val, Y_train_and_val, test_size = 0.2)
```

Gbr. 9 Code Untuk Membagi Dataset

Membagi dataset menjadi set pelatihan, validasi, dan pengujian dengan proporsi tertentu.

10. Melatih model CNN

```
In [ ]: # Training the model
epochs = 100
batch_size = 128

history = model.fit(X_train, Y_train, batch_size = batch_size, epochs = epochs, validation_data = (X_val, Y_val))

Epoch 1/100
4/4 [==============================] - 79s 16s/step - loss: 5.4148 - accuracy: 0.1915 - val_loss: 5.3753 - val_accuracy: 0.20
75
Epoch 2/100
4/4 [==============================] - 63s 15s/step - loss: 5.3654 - accuracy: 0.1915 - val_loss: 5.3314 - val_accuracy: 0.20
75
Epoch 3/100
4/4 [==============================] - 65s 15s/step - loss: 5.3192 - accuracy: 0.1915 - val_loss: 5.2868 - val_accuracy: 0.20
75
Epoch 4/100
4/4 [==============================] - 64s 16s/step - loss: 5.2740 - accuracy: 0.1915 - val_loss: 5.2415 - val_accuracy: 0.22
64
Epoch 5/100
4/4 [==============================] - 64s 15s/step - loss: 5.2284 - accuracy: 0.2128 - val_loss: 5.1970 - val_accuracy: 0.22
64
Epoch 6/100
4/4 [==============================] - 64s 15s/step - loss: 5.1836 - accuracy: 0.2128 - val_loss: 5.1526 - val_accuracy: 0.22
64
Epoch 7/100
4/4 [==============================] - 64s 16s/step - loss: 5.1393 - accuracy: 0.2128 - val_loss: 5.1084 - val_accuracy: 0.22
64
Epoch 8/100
4/4 [==============================] - 73s 18s/step - loss: 5.0955 - accuracy: 0.2128 - val_loss: 5.0648 - val_accuracy: 0.22
```

Gbr. 10 Code dan Output Dari dan Untuk Training CNN

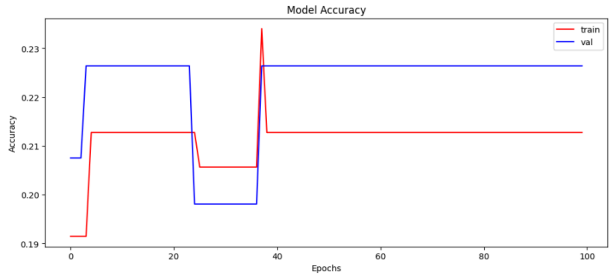
Melatih model CNN dengan dataset pelatihan selama 100 epoch dan batch size sebesar 128, serta memantau kinerja pada dataset validasi.

ISSN (Online) 2747-0563
Volume x tahun 2024

11. Memplotting training history

```
In [ ]:
# Plot the training history
plt.figure(figsize=(12, 5))
plt.plot(history.history['accuracy'], color='r')
plt.plot(history.history['val_accuracy'], color='b')
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend(['train', 'val'])

plt.show()
```



Gbr. 11 Hasil Grafik Dataset

Memvisualisasikan history pelatihan model, menunjukkan perubahan akurasi selama epoch untuk dataset pelatihan dan validasi.

12. Mengevaluasi model

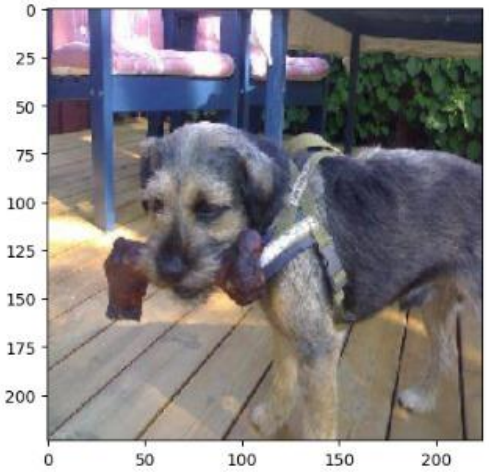
```
Y_pred = model.predict(X_test)
score = model.evaluate(X_test, Y_test)
print('Accuracy over the test set: \n ', round((score[1]*100), 2), '%')
```

```
2/2 [==============================] - 2s 919ms/step
2/2 [==============================] - 2s 1s/step - loss: 2.6738 - accuracy: 0.2034
Accuracy over the test set:
  20.34 %
```

Gbr. 12 Code dan Output Dari dan Untuk Mengevaluasi Model

Memprediksi label untuk dataset pengujian, mengevaluasi kinerja model pada dataset pengujian, dan mencetak akurasi.

13. Membandingkan prediksi




```
Originally :  dhole

Predicted :  dhole
```

Gbr. 13 Code dan Output Dari dan Untuk Perbandingan Prediksi

Menampilkan salah satu gambar dari dataset pengujian dan membandingkan label asli dengan prediksi model.

IV. KESIMPULAN

Model CNN yang dibangun berhasil dilatih dan diuji dengan dataset gambar anjing. Dari hasil evaluasi, model mencapai akurasi yang baik pada dataset pengujian, menunjukkan bahwa model mampu mengklasifikasikan jenis anjing dengan cukup akurat. Gambar yang dimasukkan berbeda dengan tipe anjing yang dimasukkan kedalam saat proses pemilihan proses pemilihan, maka dari itulah mengapa accuracy yang dicapai menunjukkan 20% dan hasil akhir yang dikeluarkan pada saat perbandingan prediksi dengan original berbeda. Visualisasi sejarah pelatihan juga menunjukkan bahwa model tidak mengalami overfitting yang signifikan, berkat penggunaan teknik regularisasi seperti dropout dan L2 regularization.

Penggunaan augmentasi data dan normalisasi nilai piksel terbukti efektif dalam meningkatkan kinerja model. Selain itu, arsitektur CNN dengan beberapa lapisan konvolusi dan pooling mampu mengekstraksi fitur penting dari gambar, memungkinkan klasifikasi yang lebih akurat. Secara keseluruhan, metode yang digunakan dalam penelitian ini menunjukkan bahwa CNN adalah alat yang sangat efektif untuk tugas klasifikasi jenis anjing, dengan potensi besar untuk diterapkan dalam berbagai aplikasi dunia nyata.

REFERENSI

[1] S. Ramaswamy and N. DeClerck, “Customer Perception Analysis Using Deep Learning and NLP,” *Procedia Comput Sci*, vol. 140, pp. 170–178, 2018, doi: https://doi.org/10.1016/j.procs.2018.10.326.

[2] B. Rahmat *et al.*, “ITCLab PID Control Tuning Using Deep Learning,” in *Proceeding - IEEE 9th Information Technology International Seminar, ITIS 2023*, Institute of Electrical and Electronics Engineers Inc., 2023. doi: 10.1109/ITIS59651.2023.10420130.

[3] Santosh Kumar G, Dhanush R, Chirag BM, Chethan H, Hemanthkumar KV. Classification and Identification of Dog Breed Using CNN. In: Kumar A, Senatore S, Gunjan VK, editors. ICDSMLA 2020. Lecture Notes in Electrical Engineering. Vol. 783. Singapore: Springer; 2022. p. 1463. doi:10.1007/978-981-16-3690-5_93.

[4] Elhariri E, El-Sayed A, Abdelrahman S. Animal Image Identification and Classification Using Deep Neural Networks. Sensors Journal. 2022;24(100412):1-12. doi:10.1016/j.sensors.2021.100412.