Training Data and Train Model Dengan MNIST Dataset Library Tensorflow

Alif Sita Maharani
Program Studi :*Teknik Informatika*
Universitas Darussalam Gontor,
Ngawi, Jawa Timur,Indonesia
(alifsita@mhs.unida.gontor.ac.id)

***Abstract*—Tensorflow merupakan salah satu package Library yang dimiliki oleh Phyton. Di dalam Package Library Teksoflow memiliki berbagai macam dataset, salah satunya adalah MNIST. MNIST adalah salah satu dataset bawaan dari Library Tensorflow. MNIST dataset ini merupakan dataset yang berisi tentang gambar dari tulisan tangan angka. Dataset yang mudah digunakan untuk training data pengujian di bidang Mechine Learning dan Image Processing . Merupakan dataset yang digunakan untuk membandingkan dnegan tingkat kesalahan terendah dalam literatur(*Abstract*)**

Keywords—MNIST Dataset, TensorFlow, Machine Learning(*key words****)***

I. INTRODUCTION (*HEADING 1*)

Dalam perkembangan teknolgi yang semakin berkemban ini selalu memiliki terobosan baru dalam berbagai macam bidangnya. Salah satunya adalah dalam bidang deep pearning dan pembelajaran mesin. Machine Learning adalah sebuah mesin yang dikembangkan untuk bisa belajar dengan sendirinya tanpa adanya arahan dari sang pengguna. Machine learning ini bermula pada awal abad ke-20, dimana seorang penemu Spanyol, Torres y Quevedo, membuat sebuah mesin learning seteleh ditemukannya komputer digital. Kata Machine Learning itu sendiri adalah proses dari sebuah komputer untuk belejar dari data. Tanpa adanya data, sebuah device tidak akan dapat melakukan apapun. Machine Learning merupakan sebagian dari ilmu keserdasan buatan atau yang sering dikenal juga dengan Artificial Intellegence (AI). Konsep dasar dari machine learning ini sendiri adalah pada pembangunan sistem agar dapat belejar sendiri tanpa memerlukan program yang di buat leh manusia berulang kali.

Tensorflow adalah sebuah library yang dikembangkan oleh google dan merupakan salah satu library yang paling populer serta banyak digunakan untuk mengembangkan dan menerapkan metode Machine Learning atau dalam algoritma lain yang memiliki banyak operasi matematika. MNIST data set (Modified National Institute of Standards and Technology database) adalah kumpulan dataset yang terdiri dari angka 0 sampai 9 yang ditulis oleh tangan. MNIST data merupakan data yang sering digunakan sebagai benchmark apakah sebuah model dapat mengenali atau melakukan klasifikasi terhadap angka – anka tersebut.

II. METODE

A. *Pengumpulan data*

Sebelum melakukan penelitian ini, pengumpulan data merupakan langkah awal yang sangat penting dalam hal penelitian. Data yang diperolah adalah dataset dari library tensorflow. Yaitu dataset MNIST yang merupakan dataset yang berupa gambar angka tulisan tangan dengan berbagai macam tulisan tangan. MNIST merupakan satu dari dataset bawaan tensorflow.

B. *Metode yang diusulkan*

Pada kali ini metode yang diusulkan menggunakan metode kuantitatif. Dimana metode ini menggunakan jenis data yang berbrntuk numerik atau sistem angka.

III. HASIL DAN PEMBAHASAN

Train data dan train model dengan MNIST dataset kali ini menggunakan aplikasi *Google Colab* dengan bahasa pemrograman *Python* dengan library *Tensorflow.*

A. *Installasi Library Tensorflow dan MNIST dataset*

Sebelum menggunakan Google Colab, tahap pertama yang dilakukan adalah melakukan installasi untuk library Tensorflow dan juga data set MNIST. Dengan mengetikkan beberapa baris kode dibawah ini :

```
import numpy as np
import matplotlib.pyplot as plt
import keras
from keras.models import Sequential
from keras.datasets import mnist
from keras.layers import Dense
from tensorflow.keras.optimizers
import Adam
from tensorflow.keras.datasets import
mnist
import random
```

B. *Import Data dari MNIST dataset*

Setelah melakukan instalasi pada library dan dataset, tahap selanjutnya adalah melakukan import pada setiap gambar pada dataset. Import data dapat dilakukan dengan mengetikkan kode dibawah ini :

```
(X_train, y_train), (X_test,
y_test) = mnist.load_data()
```

C. Mengecek Jumlah Dataset dan Membagi Menjadi 2

Setelah melakukan import, kita megecek jumlah data yang ada daam database. Cara untuk menlihat berpa banyak data dalam suatu database dengan mengetikkan kode berikut :

```
print(len(train_data))
print(len(test_data))
```

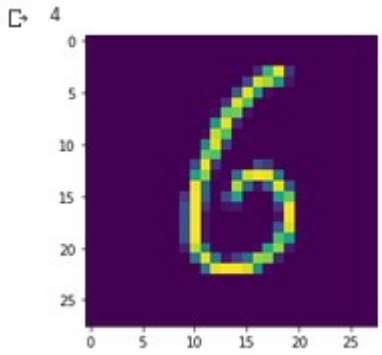
Setelah mengecek jumlah data dalam database MNIST, selanjutnya adalah membgi jumlah data menjadi 2 bagiian untuk mempermudah dalam melakukan train pada data, dengan mengetikkan kode dibawah ini :

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape[0])
```

D. Mengecek Salah Satu Gambar dan Melakukan Validasi

Tahap selanjutnya adalah memastikan gambar sudah masuk dengan mengecek salah satu gambar, lalu melakukan validasi terhadap gambar yang telah ada. Untuk melakukan pengecekan terhadap gambar, dapat mengetikkan kode dibawah ini :

```
import matplotlib.pyplot as plt
plt.imshow(train_data[18])
print(train_labels[20])
```



Gambar 1.1 Example data

Sedangkan untuk melakukan validasi, dapat mengetikkan kode dibawah ini :

```
assert(X_train.shape[0] ==
y_train.shape[0]), "The number of
images is not equal .."
assert(X_test.shape[0] ==
y_test.shape[0]), "The number of
images is not equal .."
assert(X_train.shape[1:] == (28, 28)),
"The dimension of the images are not
28x28"
assert(X_test.shape[1:] == (28, 28)),
"The dimension of the images are not
28x28
```

E. Mengambil Example Dari Setiap Data

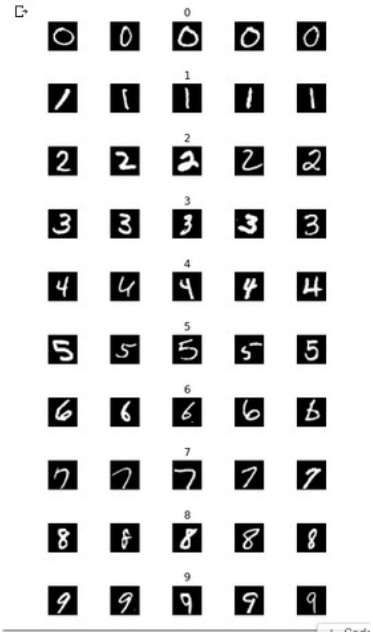
untuk mengambil beberapa sample dari data dapat dilakukan dengan mengetikan kode berikut :

```
num_of_samples = []

cols = 5 #We will select 5 random
images
num_of_classes = 10 #each digit total:
10

fig, axs =
plt.subplots(nrows=num_of_classes,
ncols=cols,
figsize=(5, 10))
fig.tight_layout()
for i in range(cols):
for j in range(num_of_classes):
x_selected = X_train[y_train == j]
axs[j]
[i].imshow(x_selected[random.randint(0
, len(x_selected -1)),
:, :],
cmap=plt.get_cmap('gray'))
axs[j][i].axis("off")
if i==2:
axs[j][i].set_title(str(j))
num_of_samples.append(len(x_selected))
```

maka akan muncul hasil seperti dibawah ini :



Identify applicable funding agency here. If none, delete this text box.

Gambar 1.2 hasil dari pengambilan sample random

F. Input Data KeDalam Diagram

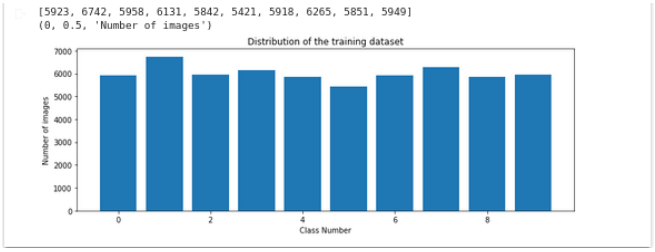
untuk memasukkan Input data kedalam diagram dapat dilakukan dengan kode berikut :

```
print(num_of_samples)
plt.figure(figsize=(12, 4))
plt.bar(range(0, num_of_classes),
num_of_samples)
plt.title("Distribution of the
training dataset")
plt.xlabel("Class Number")
plt.ylabel("Number of images")
```

```
[5923, 6742, 5958, 6131, 5842, 5421,
5918, 6265, 5851, 5949]

(0, 0.5, 'Number of images')
```

maka akan keluar hasil dalam bentuk diagram seperti gambar berikut :



Gambar 1.3 diagram

G. Penggunaan Fungsi to_categorical, Mengubah Bentuk Array

Selanjutnya adalah menggunakan fungsi to_categorical. Karena memiliki hasil multiclass dengan nilai 0-9 (10) nilai dengan mengetikkan kode berikut :

```
masing2 gambar dibagi dengan 255
intesitasnya
X_train = X_train/255
X_test = X_test/255
#we must change the shape of the
images to 1d array(28*28)
#for multipication 1*784
num_pixels = 784
X_train =
X_train.reshape(X_train.shape[0],
num_pixels)
X_test =
X_test.reshape(X_test.shape[0],
num_pixels)
print(X_train.shape)
```

Setelah melnggunakan fungsi to categorical adalah menubag bentuk array. Karena perkalian matriks yang mengharuskan mengubah bentuk array. Kita akan membetuk array 28 x 28 menjadi 1 x 728.

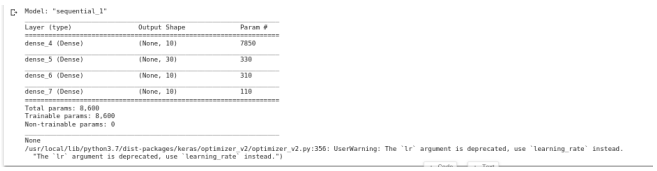
```
def create_model():
model = Sequential()
model.add(Dense(10, input_dim =
num_pixels, #num_pixels: 784
activation = 'relu'))
model.add(Dense(30,
activation='relu'))
model.add(Dense(10,
activation='relu'))
model.add(Dense(num_of_classes,
activation='softmax'))
model.compile(Adam(lr=0.01), #lr:
learning rate
loss='categorical_crossentropy', #loss
function
metrics=['accuracy'])
return model
```

H. Membuat Layer RELU

fungsi dari membuat layar RELU ini sendiri adalah untuk memberikan hasil yang lebih bagus. Untuk membuat layar RELU dapat digunakan kode berikut :

```
model = create_model()
print(model.summary())
```

maka akan menampilkan hasil seeperti berikut :



Gambar 1.4 layar RELU

I. Train a Model dan Konversi ke Greyscale

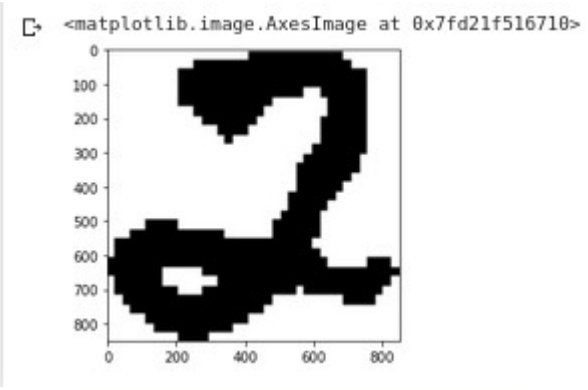
Setelah mengetikkan semua kode diatas, tahap selanjutnya adalah melakukan pelatihan terhadap model dengan gambar yang didapat dari URL dan mengubahnya kedalam bentuk greyscale dengan kode berikut :

```
import requests
from PIL import Image

url =
'https://www.researchgate.net/profile/
Jose_Sempere/publication/221258631/
figure/fig1/
AS:305526891139075@1449854695342/
Handwritten-digit-2.png'
response = requests.get(url, stream =
True)
img = Image.open(response.raw)

plt.imshow(img)
```

maka gambar yang akan muncul adalah :

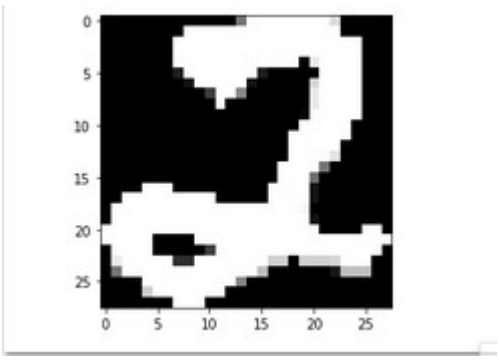


Gambar 1.5 gambar sebelum greyscale

sedangkan untuk mengubah gambar kedalam bentuk greyscale adalah dengan kode sepeti berikut :

```
import cv2
img_array = np.asarray(img)
resized = cv2.resize(img_array, (28,
28 ))
gray_scale = cv2.cvtColor(resized,
cv2.COLOR_BGR2GRAY) #(28, 28)
image = cv2.bitwise_not(gray_scale)
plt.imshow(image,
cmap=plt.get_cmap('gray'))
print(image)
```

maka gambar yang akan muncul adalah sebagai berikut :



Gambar 1.6 gambar setelah greyscale

IV. KESIMPULAN

Proses training dan train a model dengan dataset dapat dilakukan dengan berbagai macam dataset. Salah satu dataset yang digunakan kali ini adalah dataset MNIST. Dimana dataset MNIST merupakan salah satu dataset bawaan dari library TensorFlow dalam bahasa pemrograman Python. MNIST terdiri dari nilai 0-255 sehingga sebuah train data dapat dilakukan dengan baik. Yang terdiri dari 60.000 dan 10.000 data yang berbeda.

REFERENCES

[1] *https://medium.com/@afozbek_/how-to-train-a-model-with-mnist-dataset-d79f8123ba84*

[2] https://www.python-course.eu/neural_network_mnist.php

[3] https://www.tensorflow.org/datasets/keras_example

[4] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiUipXxx73zAhXFdn0KHb9xDkgQFnoECAQQAw&url=https%3A%2F%2Fiteba.ac.id%2Fblog%2Fperbedaan-metode-penelitian-kualitatif-kuantitatif-gabungan%2F&usg=AOvVaw29kecNOOmQZuqDMFN4ZLnw

[5] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiZouf3tr3zAhUF93MBHWMOC5kQFnoECAMQAw&url=https%3A%2F%2Fmedium.com%2F%40samuelsena%2Fpengenalan-deep-learning-part-6-deep-autoencoder-40d79e9c7866&usg=AOvVaw3LnLB55Kd3Ig1Cf7Zr5psI

[6] https://www.python-course.eu/neural_network_mnist.php

[7] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjzvJ2Fsr3zAhWs63MBHVHrCn4QFnoECAcQAw&url=https%3A%2F%2Fcymed.id%2Fsejarah-machine-learning%2F&usg=AOvVaw2bwJROmR_k45B_Kh_t5hdf

[8] https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjjwtKetL3zAhUWOSsKHZICCrkQFnoECAQQAQ&url=https%3A%2F%2Fdqlab.id%2Fbelajar-data-science-pahami-tensflow&usg=AOvVaw0csdF6NGBB4XqWsAXyb185

[9] https://www.youtube.com/watch?v=G6fKnx7u1Qk&t=28s