**UNIVERSITAS GUNADARMA**

**PRAKTIKUM PENGANTAR KECERDASAN BUATAN**

MANUAL BOOK

***“Visualiasi wisata di dunia”***

Nama : Nur Irfan Eka Fadhilah

NPM : 55415187

Fakultas : Teknologi Industri

Jurusan : Teknik Informatika

(PJ) : Thomi Akbar

**Ditulis Guna Melengkapi Sebagian Syarat**

**Praktikum Pengantar Kecerdasan Buatan Jenjang S1**

**Universitas Gunadarma**

**2017**

**DAFTAR ISI**

1. Daftar Isi 1
2. Bab I Pendahuluan

Latar Belakang 2

Tujuan 3

1. Bab II Pembahasan

Optimasi 4

1. Bab III Perancangan

3.1 Import dan menelurusi data 6

1. Bab IV Kesimpulan dan Saran

Kesimpulan 13

Saran 13

1. Bab V Output

Output Program...................................................................................................14

1. Daftar Pustaka 16

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Banyak para penyedia jasa travel yang melupakan betapa pentingnya mengoptimasi halaman website mereka. Padahal, optimasi merupakan menjadi faktor penting dalam untuk keberhasilan website mereka dapat di cari dalam search engine. Walaupun memakan/membutuhkan waktu yang sangat lama dan diperlukan ketelitian dan kesabaran khusus, mengoptimasi halaman website tidak boleh dipandang remeh. Banyak faktor-faktor yang menyebabkan dan bukan hanya dari search engine saja, oleh karena itu dibutuhkan program untuk optimasi gambar dengan menggunakan python.

* 1. **Tujuan**

Untuk memenuhi kewajiban agar dapat mengikuti ujian praktikum Pengantar Kecerdasan Pembuatan.

**BAB II**

**PEMBAHASAN**

**2.1 Optimasi**

Selama ini bagaimana google melakukan pengindeksan web-web ternyata bukan hanya dari artikel saja, atau kata kunci(keyword) yang ditargetkan. Ternyata Google image search(pencarian berdasarkan gambar) jadi sebaiknya gambar harus di optimasi secara maksimal.

Gambar juga di index google dihalaman pencarian gambar, yang selanjutnya akan tertuju kedalam halaman web atau blog yang sesuai dengan gambar yang dipilih. Gambar mempunyai peranan yang sangat penting dalam penampilan web/blog.

**BAB III**

**IMPLEMENTASI**

**3.1 Mengimport library dan menelusuri data**

Hal pertama daalam membuat program ini adalah menulis atau memanggil library apa saja yang dibutuhkan dalam membuat aplikasi untuk optimasi gambar, pada program kali ini menggunakan beberap library tensorflow, cv2, os, numpy, matplotlib, dan random.

**import tensorflow as tf #untuk tensorflow**

**import cv2 #untuk opencv(image processing)**

**import os #untuk ambil file dari local disk**

**import numpy as np #untuk matriks**

**import matplotlib.pyplot as plt**

**import random**

Langkah selanjutnya adalah menulis perintah untuk mengambil/menelusuri direktori dari file yang ingin di optimasi dan fungsi perulangan di gunakan untuk mengambil beberapa data sebanyak perulangan itu dibuat, dan di program kali ini yang akan diambil atau dipanggil hanya gambar yang berformat **.png**.

def load\_data(data\_directory):

directories = [d for d in os.listdir(data\_directory)

if os.path.isdir(os.path.join(data\_directory, d))]

labels = []

images = []

for d in directories:

label\_directory = os.path.join(data\_directory,d)

file\_names = [os.path.join(label\_directory, f)

for f in os.listdir(label\_directory)

if f.endswith(".png")]

for f in file\_names:

images.append(cv2.imread(f))

labels.append(int(d))

return images, labels

**3.2 Jumlah layer untuk Optimasi**

Pada program kali ini harus mendefinisikan berapa jumlah layer yang akan digunakan untuk optimasi, semakin banyak layer yang digunakan maka semakin jumlah loss saat program dijalankan.

def neural\_net(x):

layer\_1 = tf.layers.dense(x, 700, activation=tf.nn.relu)

layer\_2 = tf.layers.dense(layer\_1, 800, activation=tf.nn.relu)

layer\_3 = tf.layers.dense(layer\_2, 850, activation=tf.nn.relu)

out\_layer = tf.layers.dense(layer\_3, 900)

return out\_layer

**3.3 Directory dan Dimensi**

Setelah itu mendefinisikan lokasi(direktori) dari file yang ingin di optimasi, pada program kali ini file-file yang ingin di opmtimasi ditempatkan ke dalam direktori "D:\Python L\Python Library + Dataset" dengan nama folder Place. Setelah itu dibuat dimensi untuk gambar yaitu 30x30dan membuat array dengan nama array images30.

**ROOT\_PATH = "D:\Python L\Python Library + Dataset" #Direktori untuk library yang akan dibaca**

**train\_data\_directory = os.path.join(ROOT\_PATH, "Place") #Nama folder yang berisi library**

**images, labels = load\_data(train\_data\_directory) #untuk meload data library**

**dim = (30,30) #dimensi dari gambar yang ada di library**

**images30 = []**

**3.4 Image Processing**

Perintah di bawah adalah untuk melakukan melakukan pemrosesan gambar(image) dan merubah gambar-gambar tersebut menjadi warna gray .

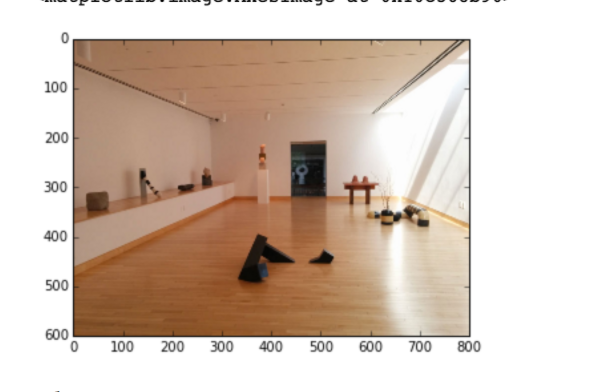
for image in images:

gray\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

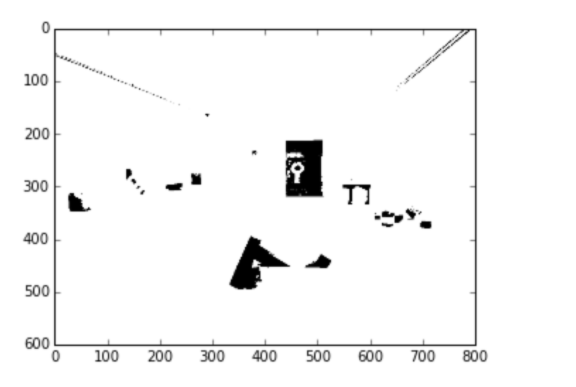
images30.append(cv2.resize(gray\_image, dim,

interpolation=cv2.INTER\_CUBIC))

Sebelum dilakukan Image Processing:

****

Setelah dilakukan Image Processing:



tf.placeholder membuat variabel placeholder yang kita masukkan ke ketika dieksekusi pada saat train atau test time.

x = tf.placeholder(dtype = tf.float32, shape = [None, 30, 30])

y = tf.placeholder(dtype = tf.int32, shape = [None])

Kode dibawah digunakan untuk melakukan umpan data ke dalam TensorFlow diluar model yang dibuat dan disimpan ke dalam variabel **x** dan **y** dimana tipe data **x** adalah float dan tipe data **y** integer dan variabel **images\_flat**  berguna sebagai inputan, dan sebagai rangsangan dalam program neural network, variabel **logits** digunakan untuk mengakses method **neural\_net** dimana parameternya adalah dari variabel **image\_flat**, **tf.nn.relu** digunakan untuk menghitung banyaknya jumlah rangsangan, sedangkan variabel loss untuk menghitung banyaknya error di jaringan neural dan berguna untuk menghitung probabilitas label dalam program.

**images\_flat = tf.contrib.layers.flatten(x) # fungnsinya array 1 dimensi ini dijadikan layar input akan lari ke baris 24**

**logits = neural\_net(images\_flat)**

**loss = tf.reduce\_mean(tf.nn.sparse\_softmax\_cross\_entropy\_with\_logits(**

**logits = logits, labels = y)) # digunakan sebagai untuk mengetahui baik buruknya hasil program bekerja, semakin kecil nilainya berati programnya semakin bagus**

Kode dibawah digunakan untuk membuat variabel **train\_op** untuk meminimalisir error di softmax yang bernilai 0.001 yang dan juga variabel **correct\_pred** digunakan untuk memprediksi kebenaran, lalu variabel **accuracy** digunakan untuk mendapatkan perhitungan dari variabel **correct\_pred** yang bertipe data float dan mengecilkan nilai rata-ratanya.

**tf.set\_random\_seed(1234) dan itu digunakan untuk melakukan operasi pengacakan terhadap data yang diambil dalam bentuk grafik.**

**train\_op = tf.train.AdamOptimizer(learning\_rate=0.0001).minimize(loss) #learning rate terlalu kecil jika melihat inputan baru, dia ttp ingat inputan lama**

**correct\_pred = tf.argmax(logits, 1) # untuk simpen benar**

**accuracy = tf.reduce\_mean(tf.cast(correct\_pred, tf.float32)) #untuk simpen akurasi belajar program**

**tf.set\_random\_seed(1234) # untuk mengambil hasil inputan secara acak**

Kode dibawah digunakan untuk menginisiasi atau mendeklarasi objek **sess,** dimana ketika objek **sess** dijalankan maka objek **sess** ini bertanggung jawab untuk menginisiasi semua variabel di dalam blok program tersebut, lalu dlakukan perulangan variabel **i** sebanyak 800 kali, ketika dilakukan perulangan maka objek sess akan mendapatkan array value dari **train\_op**, array value **loss**, dan array value **accuracy** dan disimpan ke dalam variabel **\_** , variabel **loss\_value**, dan variabel **acc**.

**with tf.Session() as sess:**

**sess.run(tf.global\_variables\_initializer())**

**for i in range(450): #mengulang i dari nilai 0-450**

**\_, loss\_value, acc = sess.run([train\_op, loss, accuracy],**

**feed\_dict={x:images30, y:labels}) #\_, loss\_value, acc tempat menyimpan train**

Kode dibawah digunakan untuk sebuah pengkondisian dimana jika setiap data variabel **i** dibagi 10 sama dengan 0 maka akan mencetak Loss: yang value nya didapat dari variabel **loss\_value**, dan mencetak variabel Accuracy: dengan value nya dari variabel **acc.**

**if i % 10 == 0:**

**print("Loss: ", loss\_value, "Accuracy: ", acc) #putaran kelipatan 10, loss value berapa dan akurasinya berapa**

Kode dibawah digunakan mengambil gambar di direktori sebanyak 10 sample secara random dan disimpan ke dalam variabel **sample\_indexes**. Lalu index yang telah dibuat perulangan variabel **i** sampai **sample\_indexes** untuk mengambil gambar sebanyak value variabel **sample\_indexes** dan disimpan ke dalam variabel **sample\_images**. Dan index yang telah dibuat perulangan variabel **i** sampai **sample\_indexes** untuk mengambil text label sebanyak value variabel **sample\_indexes** dan disimpan ke dalam variabel **sample\_labels.**

**sample\_indexes = random.sample(range(len(images30)), 10) #mengambil gambar di library sebanyak 10 sample secara random..Kenapa random? karena sudah dideklarasikan random**

**sample\_images = [images30[i] for i in sample\_indexes] #index yg udh di deklarasiin diatas, diambil 10 secara random**

**sample\_labels = [labels[i] for i in sample\_indexes] #untuk nama folder digambar sebanyak 10 buah juga**

Kode diatas digunakan untuk menjalankan machine learning menebak dengan menggunakan **sample\_images**, setelah itu membuat array beserta valuenya, dimana valuenya digunakan untuk menyimpan nama-nama buat gambar. Setelah itu membuat tempat untuk meload gambar yang berukuran 10x10, dan dilakukan perulangan sepanjang value/banyaknya gambar dari variabel **sample\_images**.lalu mengambil seluruh nilai **sample\_labels** yang sesuai dengan nilai indeks yang di awal dan disimpan ke dalam variabel **truth**, dan juga mengambil seluruh nilai **predicted** yang sesuai dengan nilai indeks yang di awal dan disimpan ke dalam variabel **prediction**, dan setelah itu membuat plot/window untuk memuat gambar dan label yang diatur ke dalam 2 kolom dan 5 baris, setelah itu mematikan grafik x dan y. Setelah itu melakukan pengkondisian nilai/gambar benar maka akan berwarna hijau dan jika salah akan berwarna merah, lalu menampilkan setiap 1 data yang terdiri dari label asli dan asli ke dalam ukuran panjang 40 dan lebar 10 dan 0 sampai 1 itu berdasarkan format string labelnya sendiri, setelah itu membuat ukuran huruf menjadi 12 dan membuat warna dalam bagian-bagian yang dibuat, setelah itu menampilkan hasil gambar, setelah itu menampilikan window untuk seluruh hasil akhir keseluruhan.

**predicted = sess.run([correct\_pred], feed\_dict={x: sample\_images})[0] #ini untuk machine learning nebak (menentukan hasil prediksi) dengan menggunakan index sample images (karena hasilnya array dan = 0 makanya di tambah akhiran [0])**

**string\_labels = ['Gereja','College','eiffel','liberty','mosque','mountain','museum'] # untuk simpen nama nama dari si gambar dari si user**

**fig = plt.figure(figsize=(10, 10)) #bikin gambar berukuran 10 x 10 , gambarnya diambil dari sample images**

**for i in range(len(sample\_images)): #perulangan index**

**truth = sample\_labels[i] #sample label dimasukin ke variabel truth (label asli)**

**prediction = predicted[i] #predicted dimasukin ke prediction (untuk label prediksinya )**

**plt.subplot(5, 2,1+i) #bikin plot buat si gambar yang terdiri dari 2 kolom dan 5 baris( banyak data dalam 1 window)**

**plt.axis('off') #mematikan grafik x dan y**

**color='green' if truth == prediction else 'red' # kalo jwb prediksi bener akan warna hijau, jika prediksi salah akan berwarna merah**

**plt.text(40, 10, "Label Asli: {0}\nHasil Prediksi: {1}".format(string\_labels[truth], string\_labels[prediction]), #(0 sama 1 itu untuk menghitung variabel berdasarkan format string labelnya sendiri)**

**fontsize=12, color=color) # 40, 10 untuk koordinatnya**

**plt.imshow(sample\_images[i]) #untuk nampilin hasil labelnya**

**plt.show() #dia akan munculin hasil akhir keseluruhannya**

**BAB IV**

**KESIMPULAN DAN SARAN**

**4.1 Kesimpulan**

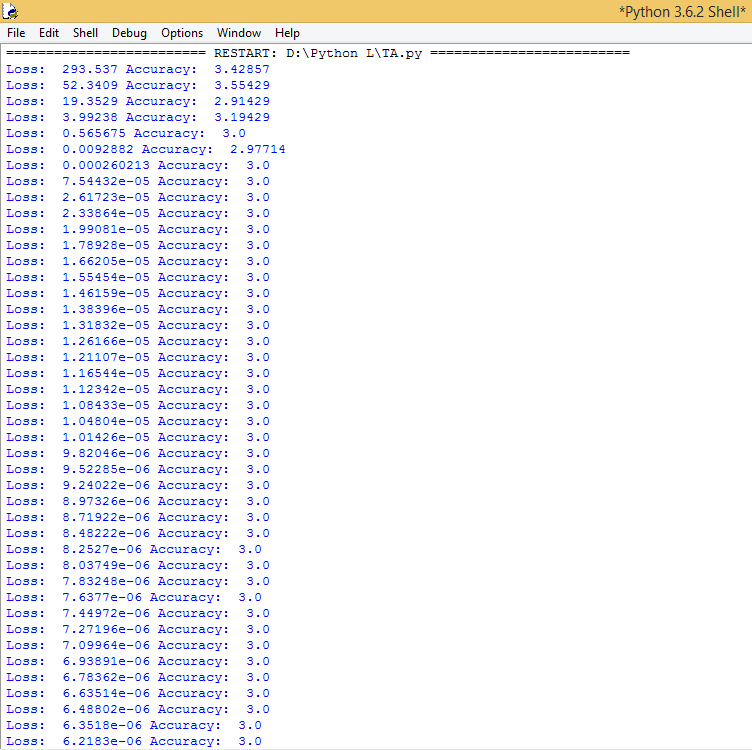
Aplikasi yang dibuat kali ini adalah aplikasi untuk melakuka optimisasi gambar sehingga bisa digunakan untuk untuk search engine salah satunya. Aplikasi ini merupakan AI(Artificial Intelligent) yaitu bisa mengenali gambar dengan baik sehingga bisa mengenali dengan mudah gambar yang telah diberika oleh kita.

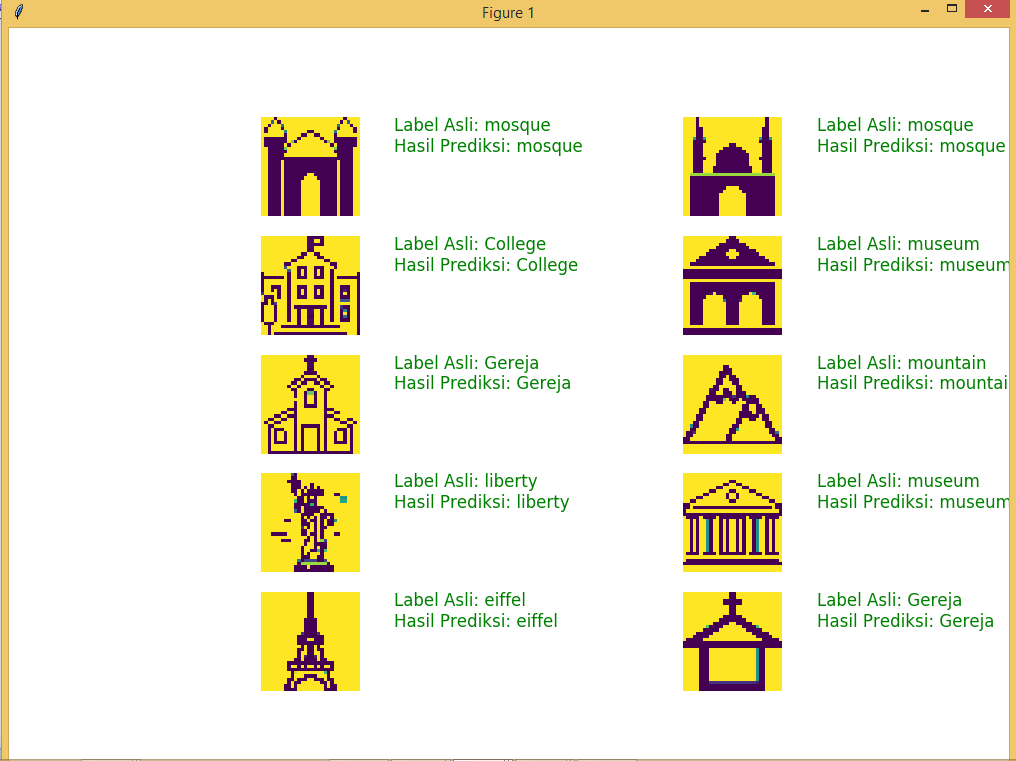
**4.2 Saran**

Semoga saja aplikasi ini bisa lebih di optimalkan lebih baik lagi dalam proses pengembangannya.

**BAB V**

**Output**





**DAFTAR PUSTAKA**