CERDAS MENGUASAI GIT

CERDAS MENGUASAI GIT
Dalam 24 Jam

Rolly M. Awangga
Informatics Research Center

Kreatif Industri Nusantara

Penulis:
Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:
M. Yusril Helmi Setyawan

Penyunting:
Syafrial Fachrie Pane
Khaera Tunnisa
Diana Asri Wijayanti

Desain sampul dan Tata letak:
Deza Martha Akbar

Penerbit:
Kreatif Industri Nusantara

Redaksi:
Jl. Ligar Nyawang No. 2
Bandung 40191
Tel. 022 2045-8529
Email : awangga@kreatif.co.id

Distributor:
Informatics Research Center
Jl. Sariasih No. 54
Bandung 40151
Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

'Jika Kamu tidak dapat
menahan lelahnya
belajar, Maka kamu harus
sanggup menahan
perihnya Kebodohan.'
Imam Syafi'i

CONTRIBUTORS

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indonesia, Bandung, Indonesia

CONTENTS IN BRIEF

DAFTAR ISI

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

Bandung, Jawa Barat
Februari, 2019

ACKNOWLEDGMENTS

Terima kasih atas semua masukan dari para mahasiswa agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk team IRC yang telah fokus untuk belajar dan memahami bagaimana buku ini mendampingi proses Intership.

R. M. A.

ACRONYMS

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association

GLOSSARY

git	Merupakan manajemen sumber kode yang dibuat oleh linus torvald.
bash	Merupakan bahasa sistem operasi berbasiskan *NIX.
linux	Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald

SYMBOLS

A Amplitude

& Propositional logic symbol

a Filter Coefficient

$\mathcal{B}$ Number of Beats

INTRODUCTION

ROLLY MAULANA AWANGGA, S.T., M.T.

Informatics Research Center
Bandung, Jawa Barat, Indonesia

Pada era disruptif saat ini. git merupakan sebuah kebutuhan dalam sebuah organisasi pengembangan perangkat lunak. Buku ini diharapkan bisa menjadi penghantar para programmer, analis, IT Operation dan Project Manajer. Dalam melakukan implementasi git pada diri dan organisasinya.

Rumusnya cuman sebagai contoh aja biar keren[**?**].

$$ABC\mathcal{D}\mathcal{E}\mathcal{F}\alpha\beta\Gamma\Delta\sum_{def}^{abc} \tag{I.1}$$

BAB 1

CHAPTER 1

BAB 2

CHAPTER 2

BAB 3

CHAPTER 3

BAB 4

CHAPTER 4

BAB 5

CHAPTER 5

BAB 6

CHAPTER 6

6.1 Damara Benedikta/ 1174012

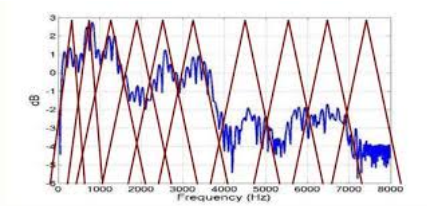
6.1.1 Teori

1. Jelaskan kenapa file suara harus dilakukan MFCC dilengkapi dengan ilustrasi atau gambar.

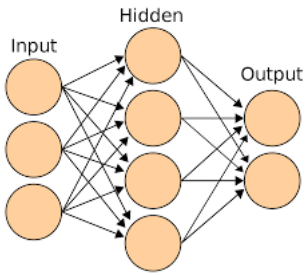
   Karena MFCCdigunakan untuk mengidentifikasi jenis suara misalkan jenis suara gendre lagu jes pop metal dan klasikal atau suara ultra sonic. Sehingga dibutuhkan penggunaan MFCC untuk memproses data tersebut agar dapat dibaca oleh manusia.

2. Jelaskan konsep dasar neural network. dilengkapo dengan ilustrasi gambar.

   Konsep neural network itu sebenarnya mengadopsi dari kemampuan otak manusia yang mampu memberikan stimulasi/rangsangan, melakukan proses, dan memberikan output. Output diperoleh dari variasi stimulasi dan proses yang terjadi di dalam otak manusia. Neural Network



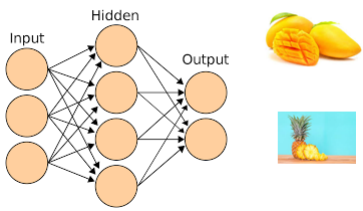
Gambar 6.1 Ilustrasi gambar metode MFCC



Gambar 6.2 Ilustrasi Konsep dasar neural network

3. Jelaskan konsep pembobotan dalam neural network. dilengkapidengan ilustrasi gambar.

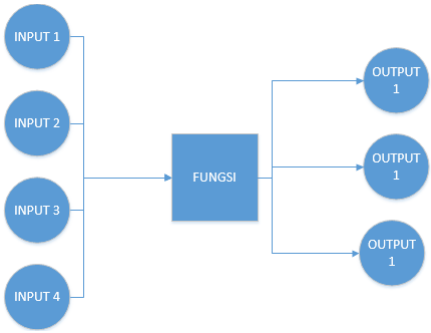
   pembobotan dalam neural network yaitu digunakan untuk membedakan objek inputan atau variabel inputan untuk AI. Dimana data inputan yang masuk adalah 2 data ”nanas” dan ”mangga” yang diolah dengan proses membandingkan data dan diolah melalui pembobotan sehingga menampilkan hasil output.



Gambar 6.3 Ilustrasi Konsep pembobotan pada neural network

4. Jelaskan konsep aktifitas dalam neural network. dilengkapi dengan ilustrasi gambar.

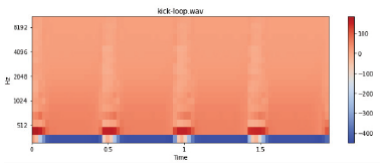
Dalam Neural Network cara aktifitas dilakukan terhadap input pada neural network inputan tersebut dimasukan kepada fungsi pada mesin sehingga di hasilkanlah output yang sesuai dengan fungsi tersebut.



Gambar 6.4 Gambar yang dibaca hasil plotnya

5. Jelaskan cara membaca hasil plot dari MFCC dilengkapi dengan ilustrasi gambar.

   cara membaca hasil ploting dari MFCC yaitu tentukan terlebih dahulu batas minimal Hz dari gelombang suara dan batas maksimal dari suara tersebut. kemudian warna yang paling pekat merupakan hasil dari pengolahan data tersebut misalkan muncul warna orange pekat di bagian bawah dan orange muda di bagian atas yang berarti suara tersebut kuat bagian basnya dan biasanya juga antara warna yang pekat tersebut ada jarak.



Gambar 6.5 Ilustrasi Cara Membaca Hasil Plot

6. Jelaskan apa itu one-hot encoding, dilengkapi dengan ilustrasi kode atau gambar.

   one-hot encoding merupakan pemberian nilai pada suatu variabel jika nilai itu positif maka nilainya satu dan jika negatif maka nilainya nol.

7. Jelaskan apa dari np.unique dan to_categorical dalam kode program, dilengkapi dengan ilustrasi atau gambar.

   fungsi dari NP.UNIQUE dalah untuk membuat data elemen menjadi nilai yang bersifat unik dalam artian (Array). Sedangkan perintah to_categorial adalah

	pop	rock	blues	rege	clasical
Lagu 1	1	0	0	0	0
Lagu 2	1	0	0	0	0
Lagu 3	0	1	0	0	0
Lagu 4	0	0	1	0	0
Lagu 5	0	0	0	1	0
Lagu 6	0	0	0	0	1
Lagu 7	0	0	0	0	1

Gambar 6.6 Ilustrasi Konsep one-hot encoding

```
>>> np . unique ([ 1 , 1 , 2 , 2 , 3 , 3 ])
array([1, 2, 3])
>>> a = np . array ([[ 1 , 1 ], [ 2 , 3 ]])
>>> np . unique ( a )
array([1, 2, 3])
```

Gambar 6.7 Ilustrasi np.unique

to_categorical

```
keras.utils.to_categorical(y, num_classes=None, dtype='float32')
```

Gambar 6.8 Ilustrasi to_categorical

untuk membuat data integer yang terdeteksi untuk diubah menjadi data matrix biner.

8. Jelaskan apa fungsi dari Sequential dalam kode program, dilengkapi dengan ilustrasi atau gambar.

   fungsi dari Sequential dari code program adalah untuk membagi data - data agar dapat dianalisis oleh sistem lebih mudah, misalkan dari data 100 dibagi prosesnya menjadi 4 yaitu 25.

Bobot 1	Bobot 2	Bobot 3	Bobot 4	Bobot 5
1-20	21-40	41-60	61-80	81-100

Gambar 6.9 Ilustrasi Konsep pembobotan pada neural network

6.1.2 Praktikum

1. Jelaskan isi dari data GTZAN Genre Collection dan data dari freesound. Buat kode program program untuk meload data tersebut untuk digunakan pada MFCC. Jelaskan arti dari perbaris kode yang dibuat (harus beda dengan teman satukelas).

   Isi data data merupakan datasets lagu atau suara yang tersiri dari 10 gendre yang di simpan kedalam 10 folder yaitu folder blues, classical, country, disco,

hiphop, jazz, metal, pop, reggae, dan rock ke sepuluh folder tersebut masing-masing berisi 100 data suara sedangkan data freesound merupakan contoh data suara yang akan di gunakan untuk menguji hasil pengolahan data tersebut dengan menggunakan metode mfcc. apakah suara dari freesound termasuk kategori jazz pop atau sebagainya ?.

```
import librosa
import librosa.feature
import librosa.display
import glob
import numpy as np
import matplotlib.pyplot as plt
from keras.layers import Dense, Activation
from keras.models import Sequential
from keras.utils.np_utils import to_categorical

# In[1]: buat fungsi mfcc untuk ngetest ajah
def display_mfcc(song):
    y, _ = librosa.load(song)
    mfcc = librosa.feature.mfcc(y)

    plt.figure(figsize=(10, 4))
    librosa.display.specshow(mfcc, x_axis='time', y_axis='mel')
    plt.colorbar()
    plt.title(song)
    plt.tight_layout()
    plt.show()
```

dapat dilihat pada kode diatas pada baris kesatu dilakukan import librosa tang digunakan untuk fungsi mfcc pada suara. pada baris kedua dilakukan import librosa featuse dan pada baris ke tiga dilakukan librosa display selanjutnya pada baris ke empat dilakukan import glob kemudian insert numpy untuk pengolahan data menjadi vektor setelah itu dilakukan import matplotlib untuk melakukan ploting setelah itu dilakukan import librari keras.

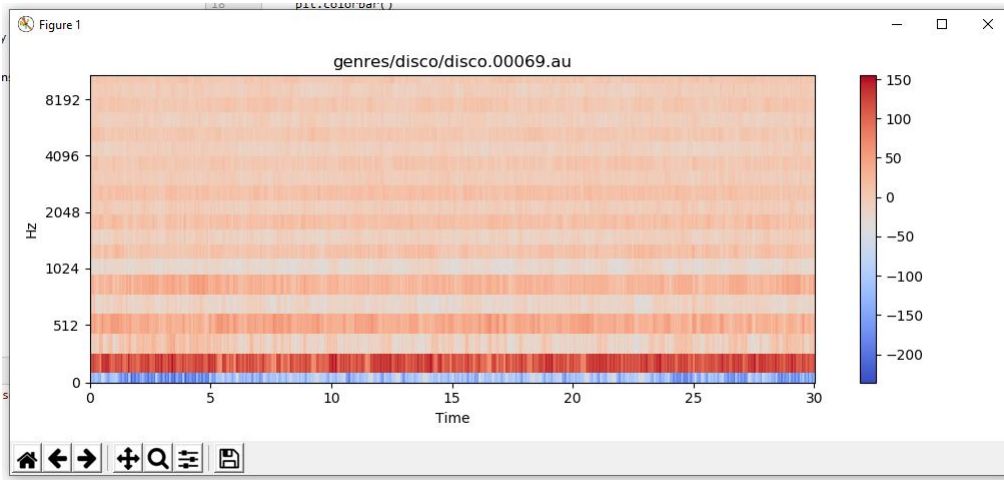
Selanjutnya yaitu membuat fungsi mfcc dengan nama display_mfcc yang didalamnya terdapat variabel y yang berisi method librosa load kemudian variabel mfcc yang berisi method librosa featurea mfcc. Setelah itu membuat flot figure dengan ukuran 10 banding 4 kemudian di isi oleh data librosa display dengan variabel x nya yaitu waktu dan y yaitu mel atau Hz kemudian melakukan plot warna setelah itu melakukan plot judul dan terakhir flot di tampilkan.

2. Jelaskan perbaris kode program dengan kata-kata dan di lengkapi ilustrasi gambar fungsi dari display_mfcc().

```
# In[2]: cek fungsi
display_mfcc('genres/disco/disco.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/blues/blues.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/classical/classical.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/country/country.00069.au')
```

```
# In[2]: cek fungsi
display_mfcc('genres/hiphop/hiphop.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/jazz/jazz.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/pop/pop.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/reggae/reggae.00069.au')
# In[2]: cek fungsi
display_mfcc('genres/rock/rock.00069.au')
```

pada baris ke dua program diatas digunakan untuk mendisplay tampilan glombang suara dari file 266093__stereo-surgeon__kick-loop-5.wav menggunakan metode mfcc dengan menggunakan fungsi display_mfcc yang telah tadi di buat pada nomer dua begitu juga pada baris ke 4 6 8 sampai ke 22 secara teksis sama menggunakan fungsi display_mfcc hanyasaja beda peyimpanan data yang akan di tampilkan atau di eksekusi. untuk contoh hasilnya dapat dilihat pada gambar **??** berikut:



Gambar 6.10 Ilustrasi gambar fungsi dari display_mfcc()

Gambar tersebut merupakan hasil dari mfcc dari salah satu gender lagu pop yang ada pada datasets yang 1000 atau terdapat dalam sepuluh folder tadi.

3. Jelaskan perbaris dengan kata-kata dan dilengkapi dengan ilustrasi gambar fungsi dari extract_features_song jelaskan kenapa data yangdiambil merupakan data 25.000 baris pertama ?

```
def extract_features_song(f):
    y, _ = librosa.load(f)

    # get Mel-frequency cepstral coefficients
    mfcc = librosa.feature.mfcc(y)
```

```
    # normalize values between -1,1 (divide by max)
    mfcc /= np.amax(np.absolute(mfcc))

    return np.ndarray.flatten(mfcc)[:25000]
```

pada baris ke tiga di definisikan nama extract_features_song yang nantinya akan di gunakan pada fungsi yang lainya kemudian dibuat variabel y dengan method librosa load setelah itu dibuat variabel baru mfcc dengan isi librosa features mfcc dengan isi variabel y tadi kemudian dibuat variabel mfcc dengan isian np.max dan variabel mfcc tadi terakhir di buat array dari data tersebut merupakan data 25000 data pertama. kenapa data 25000 pertama yang digunakan dikarenakan data tersebut digunakan sebagai data testing semakin besar data testing yang di gunakan maka semakin akurat hasil AI. tapi sebenarnya data tersebut relatif bisa lebih besar atau lebih kecil tergantung pada komputer masing masing.

4. Jelaskan Perbaris kode program dengan kata-kata dan di lengkapi ilustrasi gambar fungsi dari generate features and labels.

```
def generate_features_and_labels():
    all_features = []
    all_labels = []

    genres = ['blues', 'classical', 'country', 'disco', 'hiphop',
     'jazz', 'metal', 'pop', 'reggae', 'rock']
    for genre in genres:
        sound_files = glob.glob('genres/'+genre+'/*.au')
        print('Processing %d songs in %s genre...' % (len(
    sound_files), genre))
        for f in sound_files:
            features = extract_features_song(f)
            all_features.append(features)
            all_labels.append(genre)

    # convert labels to one-hot encoding cth blues : 1000000000
    classic 0100000000
    label_uniq_ids, label_row_ids = np.unique(all_labels,
    return_inverse=True)#ke integer
    label_row_ids = label_row_ids.astype(np.int32, copy=False)
    onehot_labels = to_categorical(label_row_ids, len(
    label_uniq_ids))#ke one hot
    return np.stack(all_features), onehot_labels
```

pada baris ke tiga merupakan pendefinisian nama fungsi yaitu generate features and labels kemudian membuat variabel baru dengan array kosing yaitu all_features dan all_labels kemudian mendefinisikan isian label untuk gendre dengan cara membuat variabel genres kemudian di isi dengan 10 gendre yang tadi setelah itu dilakukan fungsi if else dengan code for dan in setelah itu akan di buat encoding untuk data tiap tiap label contoh untuk blues 1000000000 dan untuk clasical 0100000000.

5. Jelaskan dengan kata dan praktek kenapa penggunaan fungsi generate features and labels sangat lama saat meload dataset gendre tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

   halnini menjadi lama dikarenakan mesin membaca satupersatu file yang ada pada folder dan dalam foldertersebut terdapat 100 file sehingga wajar menjadi lama ditambah lagi mengolah data yang tadinya suara menjadi bentuk vektor. berikut merupakan codenya.

```
# In[3]: passing parameter dari fitur ekstraksi menggunakan mfcc
features, labels = generate_features_and_labels()
```

Gambar 6.11 Hasil dari fungsi generate features and labels

6. jelaskan kenapa harus dilakukan pemisahan data training dan data testing sebesar 80 persen praktekan dengan kode dan tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut. untuk code nya adalah sebagai berikut yang merupakan code untuk membagi data sebanyak 80 persen untuk data training maka data musik tadi yang total jumlahnya 1000 akan di bagi dua untuk data training sebanyak 800 dan 200 untuk data testing.

```
# In[3]: fitur ektraksi
training_split = 0.8
```

7. praktekan dan jelaskan masing masing parameter dari fungsi Sequential(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

   fungsi sequential digunakan untuk mengolah data inputan sesuai dengan fungsi yang ada pada fungsi sequential pada fungsi sequential kali ini menggunakan dua fungsi sequential mengkompile data dari 100 neuron atau dari 1 folder file dengan menggunakan fungsi relu dan softmax untuk menghasilkan outputan yang sesuai dengan keriteria.

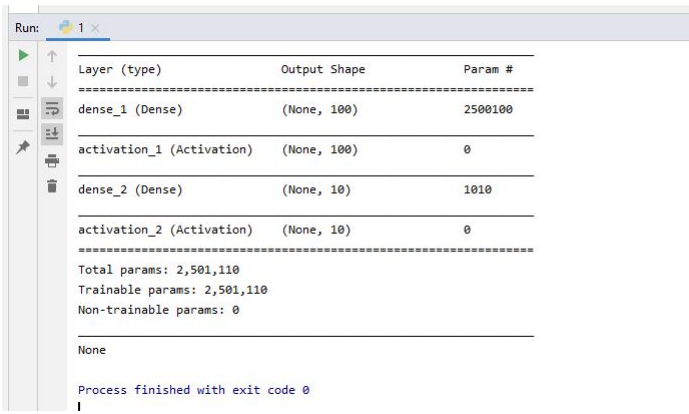
```
# In[3]: membuat seq NN, layer pertama dense dari 100 neurons
model = Sequential([
```

```
    Dense(100, input_dim=np.shape(train_input)[1]),
    Activation('relu'),
    Dense(10),
    Activation('softmax'),
    ])
```

8. praktekan dan jelaskan masing masing parameter dari fungsi compile(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

   yaitu fungsi kompile yang digunakan untuk mengetahui parameter yang digunakan dari data yang telah diolah untuk caranya dapat menggunakan codingan sebagai berikut, pada gambar tersebut memunculkan parameternya berapasaja dan total parameter yang digunakan.

```
# In[3]: fitur ektraksi
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
print(model.summary())
```

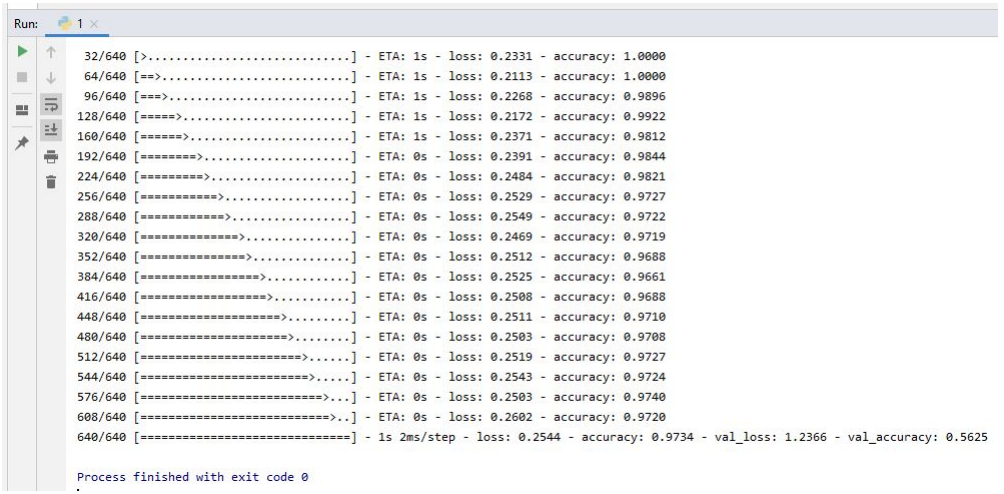


Gambar 6.12 Hasil fungsi compile

9. praktekan dan jelaskan masing masing parameter dari fungsi fit(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

   pada fungsi ini dilakukan pengolahan data dari 10 label tadi atau 10 file data sets tadi kemudian di hitung tingkat akurasi masing masing dan tingkat kegagalan atau loss data darisetiap file tersebut caranya dengan melakukan codingan berikut. pada gambar tersebut menunjukan 10 pengolahan data untuk menentukan nilai akurasi dan loss dari data tersebut dan selanjutnya dilakukan fingsi evaluasi.

```
# In[3]: fitur ektraksi
model.fit(train_input, train_labels, epochs=10, batch_size=32,
          validation_split=0.2)
```

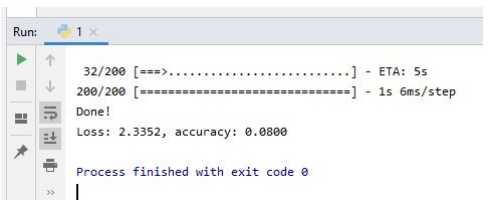


Gambar 6.13 Hasil fungsi fit

10. praktekan dan jelaskan masing masing parameter dari fungsi evaluate(). tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

    pada fungsi ini dilakukan evaluasi terhadap datayang telah di runing sebelummnya untuk lebih jelasnya dapat di lihat codingan tersebut pada codingan tersebut dilakukan evaluasi pada tingkat kegagalan dan akurasi kebenaran maka hasilnya munculkan hasil evaluasi dari 10 proses dari setiap gendre yaitu akurasi sebesar 51 persen dan loss data sebesar 1.4105 data.

```
# In[3]: fitur ektraksi
loss, acc = model.evaluate(test_input, test_labels, batch_size
    =32)
# In[3]: fitur ektraksi
print("Done!")
print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

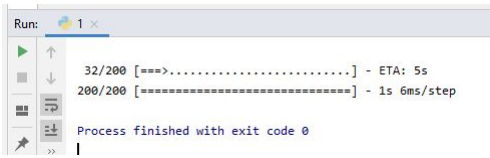


Gambar 6.14 Hasil fungsi evaluasi

11. praktekan dan jelaskan masing masing parameter dari fungsi predic tunjukan keluarannya dari komputer sendiri dan artikan maksud dari luaran tersebut.

fungsi predic merupakan fungsi untuk membandingkan tingkat akurasi pada setiap label yang sepuluh tadi maka data akan di sandingkan ke masing masing tingkat akurasinya, yang akurasinya paling tinggi maka itulah jawaban untuk setiap inputan yang dilakukan.

```
# In[3]: fitur ektraksi
model.predict(test_input[:1])
```



Gambar 6.15 Hasil fungsi prediksi

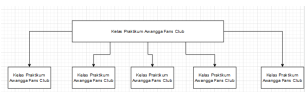
BAB 7

CHAPTER 7

7.1 1174027 - Harun Ar - Rasyid

7.1.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.



Gambar 7.1 Illustrasi Tokenizer

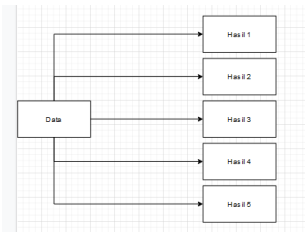
2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing **??**.dilengkapi dengan ilustrasi atau gambar.

```
    kfold = StratifiedKFold(n_splits=5)
```

```
2    splits = kfold.split(d, d['CLASS'])
3
```

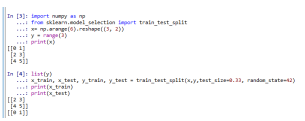
Listing 7.1 K Fold Cross Validation

Pada koding diatas terdapat variabel kfold yang didalamnya berisi parameter split yang diisikan nilai 5. hal tersebut dimaksudkan untuk membuat pengolahan data akan diulang setiap datanya sebanyak lima kali dengan atribut class sebagai acuan pengolahan datanya. Lalu kemudian akan di hasilkan akurasi dari pengulangan data tersebut sebesar sekian persen tergantung datanya



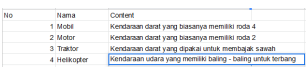
Gambar 7.2 Illustrasi K Fold Cross Validation

3. Jelaskan apa maksudnya kode program *for train, test in splits*.dilengkapi dengan ilustrasi atau gambar.
   For train digunakan untuk melakukan training atau pelatihan pada data yang sudah dideklarasikan sebelumnya. Sedangkan test in split digunakan untuk membatasi jumlah data yang akan diinputkan atau data yang akan digunakan.



Gambar 7.3 Illustrasi For train dan test in split

4. Jelaskan apa maksudnya kode program *train_content = d['CONTENT'].iloc[train_idx]* dan *test_content = d['CONTENT'].iloc[test_idx]*. dilengkapi dengan ilustrasi atau gambar.
   Maksud dari kode program tersebut adalah membaca isian kolom pada eld yang bernama CONTENT sebagai data training dan data testing untuk program



No	Nama	Content
1	Mobil	Kendaraan darat yang biasanya memiliki roda 4
2	Motor	Kendaraan darat yang biasanya memiliki roda 2
3	Traktor	Kendaraan darat yang dipakai untuk membajak sawah
4	Helikopter	Kendaraan udara yang memiliki baling - baling untuk terbang

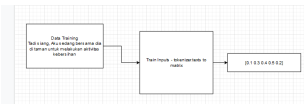
Gambar 7.4 Illustrasi penggunaan kolom Content

Jelaskan apa maksud dari fungsi *tokenizer = Tokenizer(num_words=2000)* dan *tokenizer.fit_on_texts(train_content)*, dilengkapi dengan ilustrasi atau gambar.

- tokenizer = Tokennizer(num_words=2000) digunakan untuk membaca kalimat yang telah dibuat menjadi token sebanyak 2000 kata
- fit_on_texts digunakan untuk membuat membaca data token teks yang telah dimasukan kedalam fungsi yaitu fungsi train_konten

Gambar 7.5 Illustrasi fit tokenizer dan num_word=2000

5. Jelaskan apa maksud dari fungsi *d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf')* dan *d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf')*, dilengkapi dengan ilustrasi kode dan atau gambar.
   Untuk digunakan sebagai pengubah urutan teks yang tadi telah dilakukan tkoenizer menjadi matriks yang berurutan seperti tf idf



Gambar 7.6 Illustrasi d train inputs = tokenizer.texts to matrix

6. Jelaskan apa maksud dari fungsi *d_train_inputs = d_train_inputs/np.amax(np.absolute(d_tr* dan *d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))*, dilengkapi dengan ilustrasi atau gambar.
   Fungsi tersebut digunakan untuk membagi matriks tfidf dnegan penentuan maksimum array sepanjang sumbu sehingga akan menimbulkan garis ke bawah dan ke atas yang membentuk gambar v. Lalu hasil tersebut akan dimasukkan ke variabel d train input dan d test input dengan methode absolute. Yang berarti tanpa bilangan negatif.

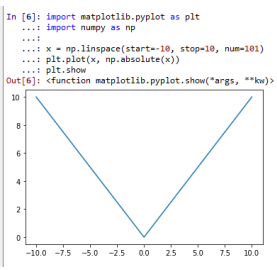
7. Jelaskan apa maksud fungsi dari *d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc* dan *d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])* dalam kode program, dilengkapi dengan ilustrasi atau gambar.
   Maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing.

8. Jelaskan apa maksud dari fungsi di listing **??**. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

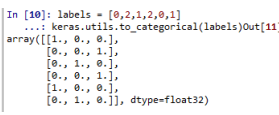
```
model = Sequential()
model.add(Dense(512, input_shape=(2000,)))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(2))
model.add(Activation('softmax'))
```

Listing 7.2 Membuat model Neural Network

model = sequential berarti variabel model berisi method sequential yang berguna untuk searching data dengan menerima parameter atau argumen kunci dengan langkah tertentu untuk mencari data yang telah diolah. Kemudian model akan ditambahkan method add dengan dense yang berarti data - data yang diinputkan akan terhubung, dengan data 612 dan 2000 data kata atau word kemudian model tersebut di masukan fungsi activation dengan rumus atau metode relu. setelah itu data akan di dropout 0.5atau dipangkas sebanyak 50 persen dikarenakan pada pohon bobot terlalu akurat terhadap data.



Gambar 7.7 Illustrasi d train inputs



Gambar 7.8 Illustrasi train outputs = np utils.to categorical

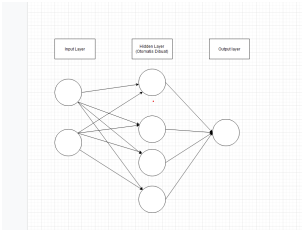
9. Jelaskan apa maksud dari fungsi di listing **??** dengan parameter tersebut.

```
model.compile(loss='categorical_crossentropy', optimizer='
adamax',
              metrics=['accuracy'])
```

Listing 7.3 Compile model

model tersebut kemudian di compile atau di kembalikan kembali fungsi nilainya yangmana akan mengembalikan fungsi nilai loss nya berapa yang diambil dari

fungsi adamax yang berberguna untuk mengetahui nilai lossnya kemudian metrics = acuracy merupakan akurasi dari nilai matrixnya. kemudian terdiri atas beberapa layer atau hiden layer. perbedaan antara deep learning dan DNN atau Deep Neural Network yaitu deep lerning merupakan pemakai algoritma dari DNN dan DNN merupakan algoritma yang ada pada deep learning.

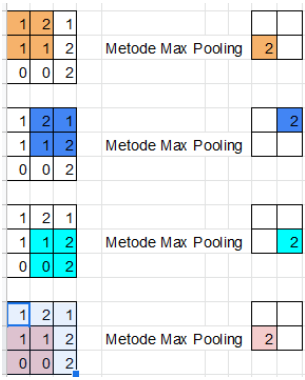


Gambar 7.9 Illustrasi Neural Network

10. Jelaskan apa itu Deep Learning
    Deep learning merupakan salah satu algoritma yang seperti Neural Network yang menggunakan meta data sebagai inputan dan mengolahnya menggunakan layer layer yang tersembunyi.

11. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning
    Deep Neural Network merupakan algoritma jaringan syaraf yang melakukan pembobotan terhadap data yang sudah ada sebagai acuan untuk data inputan selanjutnya.

12. Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1) x (NPM mod3+1) yang terdapat max pooling.(nilai 30)
    sebelum membuat ilustrasi perlu di ketahui apa itu stride, stride adalah acuan atau parameter yang menentukan pergeseran pada lter xcel. sebagai contoh nilai stride 1 yang berarti lter akan bergeser sebanyak satu xcel secara vertikal dan horizontal. selanjutnya apa itu max pooling contoh pada suatu gambar di tentukan Max Pooling dari 3 x 3 dengan stride 1 yang berarti setiap pergeseran 1 pixcel akan diambil nilai terbesar dari pixcel 3 x 3 tersebut.



Gambar 7.10 Illustrasi perhitungan stride 1 max pooling

7.1.2 Praktek

1. Jelaskan kode program pada blok # In[1]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[1]:import lib
# menimport libtari CSV untuk mengolah data ber ekstensi csv
import csv
#kemudian Melakukan import library Image yang berguna untuk dari PIL atau Python Imaging Library yang berguna untuk mengolah data berupa gambar
from PIL import Image as pil_image
# kemudian Melakukan import library keras yang menggunakan method preprocessing yang digunakan untuk membuat neural network
import keras.preprocessing.image
```

2. Jelaskan kode program pada blok # In[2]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[2]:load all images (as numpy arrays) and save their classes
#Menginisiasi variabel imgs dan classes dengan variabel array kosong
imgs = []
classes = []
#membuaka file hasy-data-labels.csv yang berada di folede HASYv2 yang di inisialisasi menjadi csvfile
with open('HASYv2/hasy-data-labels.csv') as csvfile:
    #Menginisiasi variabel csvreader yang berisi method csv.reader yang membaca variabel csvfile
    csvreader = csv.reader(csvfile)
    # Menginisiasi variabel i dengan isi 0
    i = 0
    # membuat looping pada variabel csvreader
    for row in csvreader:
```

```
        # dengan ketentuan jika i lebihkecil daripada o
        if i > 0:
            # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
            #Pembagian data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # Penambahan nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # Penambahan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

3. Jelaskan kode program pada blok # In[3]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[3]:shuffle the data, split into 80% train, 20% test
# Melakukan import library random
import random
# melakukan random pada vungsi imgs
random.shuffle(imgs)
# Menginisiasi variabel split_idx dengan nilai integer 80 persen dikali dari pengembalian jumlah dari variabel imgs
split_idx = int(0.8*len(imgs))
# Menginisiasi variabel train dengan isi lebih besar split idx
train = imgs[:split_idx]
# Menginisiasi variabel test dengan isi lebih kecil split idx
test = imgs[split_idx:]
```

4. Jelaskan kode program pada blok # In[4]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[4]:
# Melakukan import library numpy dengan inisial np
import numpy as np
# Menginisiasi variabel train input dengan np method asarray yang mana membuat array dengan isi row 2 dari data train
train_input = np.asarray(list(map(lambda row: row[2], train)))
# membuat test input input dengan np method asarray yang mana membuat array dengan isi row 2 dari data test
test_input = np.asarray(list(map(lambda row: row[2], test)))
# Menginisiasi variabel train_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data train
train_output = np.asarray(list(map(lambda row: row[1], train)))
# Menginisiasi variabel test_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data test
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

5. Jelaskan kode program pada blok # In[5]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[5]: import encoder and one hot
# Melakukan import library LabelEncode dari sklearn
from sklearn.preprocessing import LabelEncoder
# Melakukan import library OneHotEncoder dari sklearn
from sklearn.preprocessing import OneHotEncoder
```

6. Jelaskan kode program pada blok # In[6]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[6]:convert class names into one-hot encoding

# Menginisiasi variabel label_encoder dengan isi LabelEncoder
label_encoder = LabelEncoder()
# Menginisiasi variabel integer_encoded yang berfungsi untuk
    Menconvert variabel classes kedalam bentuk integer
integer_encoded = label_encoder.fit_transform(classes)
```

7. Jelaskan kode program pada blok # In[7]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[7]:then convert integers into one-hot encoding
# Menginisiasi variabel onehot_encoder dengan isi OneHotEncoder
onehot_encoder = OneHotEncoder(sparse=False)
# mengisi variabel integer_encoded dengan isi integer_encoded
    yang telah di convert pada fungsi sebelumnya
integer_encoded = integer_encoded.reshape(len(integer_encoded),
    1)
# Menconvert variabel integer_encoded kedalam onehot_encoder
onehot_encoder.fit(integer_encoded)
```

8. Jelaskan kode program pada blok # In[8]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[8]:convert train and test output to one-hot
# Menconvert data train output  mengguanakn variabel
    label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# Menconvert variabel train_output_int kedalam fungsi
    onehot_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
# Menconvert data test_output mengguanakn variabel label_encoder
    kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# Menconvert variabel test_output_int kedalam fungsi
    onehot_encoder
```

```
test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
# Menginisiasi variabel num_classes dengan isi variabel
    label_encoder dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

9. Jelaskan kode program pada blok # In[9]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[9]: import sequential
# Melakukan import library Sequential dari Keras
from keras.models import Sequential
# Melakukan import library Dense, Dropout, Flatten dari Keras
from keras.layers import Dense, Dropout, Flatten
# Melakukan import library Conv2D, MaxPooling2D dari Keras
from keras.layers import Conv2D, MaxPooling2D
```

10. Jelaskan kode program pada blok # In[10]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[10]: desain jaringan
# Menginisiasi variabel model dengan isian library Sequential
model = Sequential()
# variabel model di tambahkan library Conv2D tigapuluh dua bit
    dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang
    menggunakan data train_input
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
# variabel model di tambahkan dengan lib MaxPooling2D dengan
    ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan dengan library Conv2D 32bit dengan
    kernel 3 x 3
model.add(Conv2D(32, (3, 3), activation='relu'))
# variabel model di tambahkan dengan lib MaxPooling2D dengan
    ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan library Flatten
model.add(Flatten())
# variabel model di tambahkan library Dense dengan fungsi tanh
model.add(Dense(1024, activation='tanh'))
# variabel model di tambahkan library dropout untuk memangkas
    data tree sebesar 50 persen
model.add(Dropout(0.5))
# variabel model di tambahkan library Dense dengan data dari
    num_classes dan fungsi softmax
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan
    optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy'])
```

```
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

11. Jelaskan kode program pada blok # In[11]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[11]: import sequential
# Melakukan import library keras callbacks
import keras.callbacks
# Menginisiasi variabel tensorboard dengan isi lib keras
tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

12. Jelaskan kode program pada blok # In[12]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[12]: 5menit kali 10 epoch = 50 menit
# fungsi model titambahkan metod fit untuk mengetahui perhitungan dari train_input train_output
model.fit(train_input, train_output,
# dengan batch size 32 bit
          batch_size=32,
          epochs=10,
          verbose=2,
          validation_split=0.2,
          callbacks=[tensorboard])

score = model.evaluate(test_input, test_output, verbose=2)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

13. Jelaskan kode program pada blok # In[13]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[13]:try various model configurations and parameters to find the best
# Melakukan import library time
import time
#Menginisiasi variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # Menginisiasi variabel model Sequential
```

```
13          model = Sequential()
14          #membuat looping untuk variabel i dengan jarak dari
    hasil konvolusi.
15          for i in range(conv2d_count):
16              # syarat jika i samadengan bobotnya 0
17              if i == 0:
18                  # Penambahan method add pada variabel model
    dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat
    kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data
    shape yang di hitung dari data train.
19                  model.add(Conv2D(32, kernel_size=(3, 3),
    activation='relu', input_shape=np.shape(train_input[0])))
20                  # jika tidak
21              else:
22                  # Penambahan method add pada variabel model
    dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3
    dan fungsi aktivasi relu
23                  model.add(Conv2D(32, kernel_size=(3, 3),
    activation='relu'))
24              # Penambahan method add pada variabel model
    dengan isian method  Max pooling berdimensi 2 dengan ukuran
    fixcel 2 x 2.
25              model.add(MaxPooling2D(pool_size=(2, 2)))
26          # merubah feature gambar menjadi 1 dimensi vektor
27          model.add(Flatten())
28          # Penambahan method dense untuk pemadatan data dengan
     ukuran dense di tentukan dengan rumus fungsi tanh.
29          model.add(Dense(dense_size, activation='tanh'))
30          # membuat ketentuan jika pemangkasan lebih besar dari
     0 persen
31          if dropout > 0.0:
32              # Penambahan method dropout pada model dengan
    nilai dari dropout
33              model.add(Dropout(dropout))
34              # Penambahan method dense dengan fungsi num
    classs dan rumus softmax
35          model.add(Dense(num_classes, activation='softmax'))
36          # mongkompile variabel model dengan hasi loss
    optimasi dan akurasi matrix
37          model.compile(loss='categorical_crossentropy',
    optimizer='adam', metrics=['accuracy'])
38          # melakukan log pada dir
39          log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
    (conv2d_count, dense_size, dropout)
40          # Menginisiasi variabel tensorboard dengan isian dari
     library keras dan nilai dari lig dir
41          tensorboard = keras.callbacks.TensorBoard(log_dir=
    log_dir)
42          # Menginisiasi variabel start dengan isian dari
    library time menggunakan method time
43
44          start = time.time()
45          # Penambahan method fit pada model dengan data dari
    train input train output nilai batch nilai epoch verbose
    nilai 20 persen validation split dan callback dengan nilai
    tnsorboard.
```

```
            model.fit(train_input, train_output, batch_size=32, epochs=10,
                      verbose=0, validation_split=0.2, callbacks=[tensorboard])
            # Menginisiasi variabel score dengan nilai evaluasi dari model menggunakan data tes input dan tes output
            score = model.evaluate(test_input, test_output, verbose=2)
            # Menginisiasi variabel end
            end = time.time()
            # Menginisiasi variabel elapsed
            elapsed = end - start
            # mencetak hasil perhitungan
            print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
            results.append((conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
```

14. Jelaskan kode program pada blok # In[14]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[14]:rebuild/retrain a model with the best parameters (from the search) and use all data
# Menginisiasi variabel model dengan isian library Sequential
model = Sequential()
# variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan dengan library Conv2D 32bit dengan kernel 3 x 3
model.add(Conv2D(32, (3, 3), activation='relu'))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan library Flatten
model.add(Flatten())
# variabel model di tambahkan library Dense dengan fungsi tanh
model.add(Dense(128, activation='tanh'))
# variabel model di tambahkan library dropout untuk memangkas data tree sebesar 50 persen
model.add(Dropout(0.5))
# variabel model di tambahkan library Dense dengan data dari num_classes dan fungsi softmax
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

```
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

15. Jelaskan kode program pada blok # In[15]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[15]:join train and test data so we train the network on all data we have available to us
# melakukan join numpy menggunakan data train_input test_input
model.fit(np.concatenate((train_input, test_input)),
          # kelanjutan data yang di gunakan pada join train_output test_output
          np.concatenate((train_output, test_output)),
          #menggunakan ukuran 32 bit dan epoch 10
          batch_size=32, epochs=10, verbose=2)
```

16. Jelaskan kode program pada blok # In[16]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[16]:save the trained model
#menyimpan model atau mengeksport model yang telah di jalantadi
model.save("mathsymbols.model")
```

17. Jelaskan kode program pada blok # In[17]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[17]:save label encoder (to reverse one-hot encoding)
# menyompan label encoder dengan nama classes.npy
np.save('classes.npy', label_encoder.classes_)
```

18. Jelaskan kode program pada blok # In[18]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[18]:load the pre-trained model and predict the math symbol for an arbitrary image;
# the code below could be placed in a separate file
# mengimpport library keras model
import keras.models
# Menginisiasi variabel model2 untuk meload model yang telah di simpan tadi
model2 = keras.models.load_model("mathsymbols.model")
# mencetak hasil model2
print(model2.summary())
```

19. Jelaskan kode program pada blok # In[19]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[19]:restore the class name to integer encoder
# Menginisiasi variabel label encoder ke 2 dengan isian fungsi label encoder.
label_encoder2 = LabelEncoder()
# Penambahan method classess dengan data classess yang di eksport tadi
label_encoder2.classes_ = np.load('classes.npy')
# membuat fumgsi predict dengan path img
def predict(img_path):
    # Menginisiasi variabel newimg dengam membuay immage menjadi array dan membuka data berdasarkan img path
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path))
    # membagi data yang terdapat pada variabel newimg sebanyak 255
    newimg /= 255.0

    # do the prediction
    # Menginisiasi variabel predivtion dengan isian variabel model2 menggunakan fungsi predic dengan syarat variabel newimg dengan data reshape
    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

    # figure out which output neuron had the highest score, and reverse the one-hot encoding
    # Menginisiasi variabel inverted  denagan label encoder2 dan menggunakan argmax untuk mencari skor luaran tertinggi
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
    # mencetak prediksi gambar dan confidence dari gambar.
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

20. Jelaskan kode program pada blok # In[20]. Jelaskan arti dari setiap baris kode yang dibuat(harus beda dengan teman sekelas) dan hasil luarannya dari komputer sendiri.

```
# In[20]: grab an image (we'll just use a random training image for demonstration purposes)
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00010.png
predict("HASYv2/hasy-data/v2-00010.png")
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00500.png
predict("HASYv2/hasy-data/v2-00500.png")
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00700.png
predict("HASYv2/hasy-data/v2-00700.png")
```

7.1.3 Penanganan Error

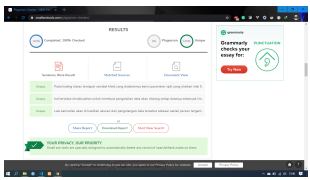
1. SS Error

```
File "D:/Kuliah/Semester 6/Kecerdasan Buatan/Bahan/src/1174027/6/1174027.py", line 9,
in <module>
    import librosa
ModuleNotFoundError: No module named 'librosa'
```

Gambar 7.11 No Module Name error

2. Jenis Error
   - No Module
3. Cara Penanganan
   Dengan cara melakukan instalasi module yang bersangkutan / menginstal library yang digunakan

7.1.4 Bukti Tidak Plagiat



Gambar 7.12 Tidak Melakukan Plagiat Pada Ch 7

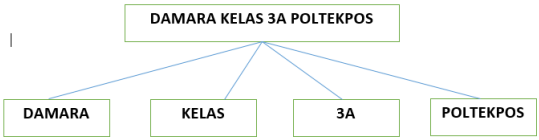
7.2 Damara Benedikta / 1174012

7.2.1 Teori

1. Jelaskan Kenapa file teks harus dilakukan tokenizer. Dilengkap dengan ilustrasi atau gambar.

   Tokenizer adalah sebuah langkah pertama yang diperlukan dalam tugas pemrosesan bahasa, seperti penghitungan kata, penguraian, pemeriksaan ejaan, pembuatan corpus, dan analisis statistik teks. Itu mengkonversi string teks Python menjadi aliran objek token, di mana setiap objek token adalah kata yang terpisah, tanda baca, nomor / jumlah, tanggal, email, URL / URI, dll. Ini juga mengelompokkan aliran token menjadi kalimat, dengan mempertimbangkan sudut kasus-kasus seperti singkatan dan tanggal di tengah kalimat. dimana proses untuk membagi kalimat menjadi beberapa teks, hal ini sangat di perlukan dalam AI karena nanti setiap teks akan di hitung bobotnya dang akan memunculkan nilai vektor sehingga teks tersebut bisa di gunakan sebagai data untuk memprediksi teks yang muncul dalam satu kalimat sedangakan proses tokenizer merupakan caramembagi bagi teks dari suatu kalimat biasanya pembagi kalimat tersebut merupakan spasi dalam suatu kalimat.

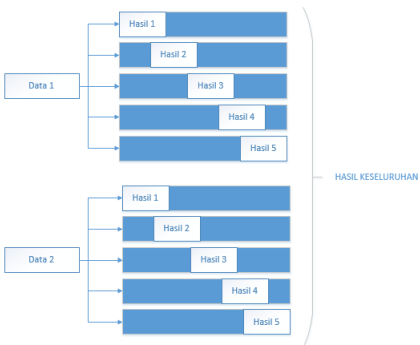
2. jelaskan konsep dasar K Fold Cross Validation dan dataset komentar youtube pada code berikut dilengkapi dengan ilustrasi gambar



Gambar 7.13 Ilustrasi Tokenisasi

```
kfold = StartifiedKFlod(n_splits=5)
splits = kfold.split(d, d['CLASS'])
```

pada codingan tersebut terdapat kfold sebagai variabel yang didalammnya terdiri dari split 5 yang berarti pengulangan terhadap pengolahan masing masing lima kali pada kasus ini terdapat data sebanyak 5 berarti ke lima data tersebut di ulang sebanyak lima kali dengan atribut class sebagai acuan pengolahan datanya kemudian akan di hasilkan akurasi dari pengulangan data tersebut sebesar sekian persen tergantung datanya.



Gambar 7.14 Ilustrasi K Fold Cross Validation

3. Jelaskan apa maksudnya kode program for train, test in split dilengkapi di lengkapi dengan ilustrasi atau gambar.

   for train di gunakan melakukan training atau pelatihan terhadap data yang telah di deklarasikan sebelumnya. sedangkan test in split di gunakan untuk diguanakan untuk membatasi jumlah data yang akan di inputkan atau data yang akan di gunakan.

4. Jelaskan apa maksud kode program traint_content =d['CONTENT'].iloc[traint_idx] dan test_content =d['CONTENT'].iloc[traint_idx]. dilengkapi dengan ilustrasi atau gambar.

```
>>> import numpy as np
>>> from sklearn.model_selection import train_test_split
>>> X, y = np.arange(6).reshape((3, 2)), range(3)
>>> X
array([[0, 1],
       [2, 3],
       [4, 5]])
>>> list(y)
[0, 1, 2]
>>> X_train, X_test, y_train, y_test = train_test_split(
...     X, y, test_size=0.33, random_state=42)
>>> X_train
array([[2, 3],
       [4, 5]])
>>> X_test
array([[0, 1]])
>>>
```

Gambar 7.15 Ilustrasi For Train dan Test in split

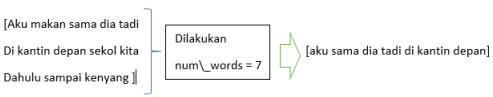
maksud dari kode program tersebut adalah membaca isian kolom pada field yang bernama CONTENT sebagai data training dan data testing untuk program tersebut.

No	Nama	KONTENT
1	Cokro	Seseorang yang berusaha
2	Edi	Anak yang patuh pada ibunya
3	Prawiro	Seorang prajurit negara indonesia
4	Ali	Orang yang berlapang dada
5	Hbibi	Dia yang di berkati akan semua tindakannya

Gambar 7.16 Ilustrasi Penggunaan kolom CONTENT

5. Jelaskan maksud dari fungsi tokenizer = Tokennizer(num_words=2000)dan tokenizer.fit_on_texts(train_konten) di lengkapi dengan ilustrasi gambar.

   fungsi tokenizer = Tokennizer(num_words=2000) digunakan untuk membaca kalimat yang telah di buat menjadi token sebanyak 2000 kata dan fingsi fit_on_texts(train_k digunakan untuk membuat membaca data token teks yang telah di masukan kedalam fungsi yaitu fungsi train_konten.

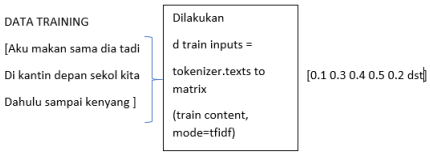


Gambar 7.17 Ilustrasi fit tokenizer dan num_word=2000

6. Jelaskan apa maksud dari fungsi d train inputs = tokenizer.texts to matrix(train content, mode=tfidf) dan d test inputs = tokenizer.texts to matrix(test content, mode=tfidf), dilengkapi dengan ilustrasi kode dan atau gambar.

   yaitu di gunakan untuk mengubah urutan teks yang tadi telah di lakukan tokenizer menjadi matriks yang berut=rutan seperti tf idf

7. Jelaskan apa maksud dari fungsi *d_train_inputs = d_train_inputs/np.amax(np.absolute(d_tr* dan *d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs))*, dilengkapi dengan ilustrasi atau gambar

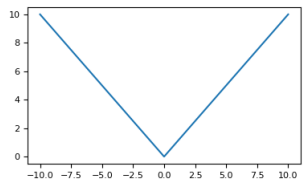


Gambar 7.18 Ilustrasi d train inputs = tokenizer.texts to matrix(train content, mode=tfidf)

fungsi tersebut digunakan untuk membagi matriks tfidf dengan dengan penentuan maksimum aray sepanjang sumbu sehingga akan menimbulkan garis ke bawah dan keatas yang membentuk gambar v kemudian hasil tersebut akan di masukan ke dalam variabel d train input dan d test input dengan methode absolute. yang berarti tanpa bilangan negatif.

```
>>> import matplotlib.pyplot as plt
```

```
>>> x = np.linspace(start=-10, stop=10, num=101)
>>> plt.plot(x, np.absolute(x))
>>> plt.show()
```



Gambar 7.19 Ilustrasi d_train_inputs = d_train_inputs/np.amax(np.absolute(d_train_inputs))

8. Jelaskan apa maksud fungsi dari d train outputs = np utils.to categorical(d['CLASS'].iloc[t dan d test outputs = np utils.to categorical(d['CLASS'].iloc[test idx]) dalam kode program, dilengkapi dengan ilustrasi atau gambar.

   maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing.

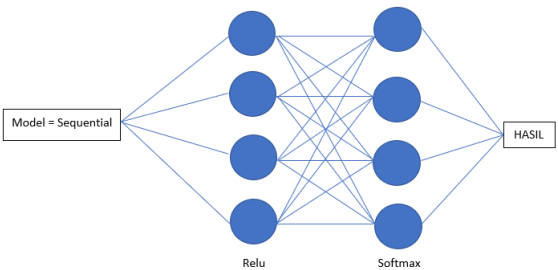
9. model = sequential berarti variabel model berisi method sequential yang berguna untuk searching data dengan menerima parameter atau argumen kunci dengan langkah tertentu untuk mencari data yang telah di olah. kemudian modeldi tambahkan metod add dengan Dense yang berarti data data yang di inputkan akan terhubung, dengan data 612 dan 2000 data kata atau word kemudian model tersebut di masukan fungsi aktivation dengan rumusa atau methode relu setelah itu data tersebut di dropout 0.5 atau di pangkas sebanyak 50 persen di karenakan pada pohon bobot terlalu akurat terhadap data. sehingga data dilakukan pemangkasan 50 persen. kemudian data tersebut di hubungkan. setelah data terse-

```
# Consider an array of 5 labels out of a set of 3 classes {0, 1, 2}:
> labels
array([0, 2, 1, 2, 0])
# `to_categorical` converts this into a matrix with as many
# columns as there are classes. The number of rows
# stays the same.
> to_categorical(labels)
array([[ 1.,  0.,  0.],
       [ 0.,  0.,  1.],
       [ 0.,  1.,  0.],
       [ 0.,  0.,  1.],
       [ 1.,  0.,  0.]], dtype=float32)
```

```
>>> type(df.iloc[0])
<class 'pandas.core.series.Series'>
>>> df.iloc[0]
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Gambar 7.20 Ilustrasi train outputs = np utils.to categorical(d['CLASS'].iloc[train

but di hubungkan maka di lakukan perhitungan dengan menggunakan fungsi softmax.



Gambar 7.21 Ilustrasi Neural Network

10. model tersebut kemudian di compile atau di kembalikan kembali fungsi nilainya yangmana akan mengembalikan fungsi nilai loss nya berapa yang diambil dari fungsi adamax yang berberguna untuk mengetahui nilai lossnya kemudian metrics = acuracy merupakan akurasi dari nilai matrixnya.

11. Jelaskan apa itu Deep Learning

    merupakan salah satu algoritma jaringan saraf tiruan yang menggunakan meta data sebagai inputan dan mengolahnya menggunakan layer layer yang tersembunyi. deep lerning memiliki suatu keunikan yaitu fitur yang dapat mengekstraksi secara otomatis.

12. Jelaskan Apa itu Deep Neural Network dan apa perbedaanya dengan Deep Learning.

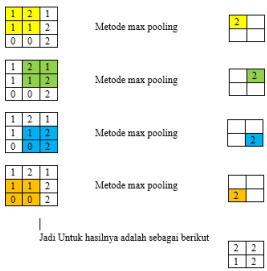
    merupakan algoritma jaringan syaraf juga yang mena akan melakukan pembobotan terhadap data yang sudah ada sebagai acuan untuk data inputan selanjutnya. kemudian terdiri atas beberapa layer atau hiden layer. perbedaan antara

deep learning dan DNN atau Deep Neural Network yaitu deep lerning merupakan pemakai algoritma dari DNN dan DNN merupakan algoritma yang ada pada deep learning.

13. Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1) x (NPM mod3+1) yang terdapat max pooling.(nilai 30)

    sebelum membuat ilustrasi perlu di ketahui apa itu stride, stride adalah acuan atau parameter yang menentukan pergeseran pada filter fixcel. sebagai contoh nilai stride 1 yang berarti filter akan bergeser sebanyak satu fixcel secara vertikal dan horizontal. selanjutnya apa itu max pooling contoh pada suatu gambar di tentukan Max Pooling dari 3 x 3 dengan stride 1 yang berarti setiap pergeseran 1 pixcel akan diambil nilai terbesar dari pixcel 3 x 3 tersebut.

    selanjutnya jawaban dari soal ini yaitu menggunakan stride 1 dengan ketentuan max pooling.



Gambar 7.22 Hasil perhitungan stride 1 max pooling

7.2.2 Pemrograman

Penjelasan setiap baris code dari nomer 1 sampai 20 bisa di lihat pada bagian coo-ment coding yang berwarna hijau.

1. Jelaskan kode program pada blok In[1].

```
# In[1]:import lib
# menimport libtari CSV untuk mengolah data ber ekstensi csv
import csv
#kemudian mengimport librari Image yang berguna untuk dari PIL
    atau Python Imaging Library yang berguna untuk mengolah data
    berupa gambar
from PIL import Image as pil_image
# kemudian mengimport librari keras yang menggunakan method
    preprocessing yang digunakan untuk membuat neutal network
import keras.preprocessing.image
```

   Untuk hasilnya dapat di lihat pada gambar berikut.

2. Jelaskan kode program pada blok In[2].

Gambar 7.23 hasil kode program

```
# In[2]:load all images (as numpy arrays) and save their classes
#membuat variabel imgs dengan variabel kosong
imgs = []
#membuat variabel classes dengan variabel kosong
classes = []
#membuaka file hasy-data-labels.csv yang berada di folede HASYv2 yang di inisialisasi menjadi csvfile
with open('HASYv2/hasy-data-labels.csv') as csvfile:
    #membuat variabel csvreader yang berisi method csv.reader yang membaca variabel csvfile
    csvreader = csv.reader(csvfile)
    # membuat variabel i dengan isi 0
    i = 0
    # membuat looping pada variabel csvreader
    for row in csvreader:
        # dengan ketentuan jika i lebihkecil daripada o
        if i > 0:
            # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
            # neuron activation functions behave best when input values are between 0.0 and 1.0 (or -1.0 and 1.0),
            # so we rescale each pixel value to be in the range 0.0 to 1.0 instead of 0-255
            #membagi data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # menambah nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # menambahkan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

Untuk hasilnya dapat di lihat pada gambar berikut.

3. Jelaskan kode program pada blok In[3].

```
# In[3]:shuffle the data, split into 80% train, 20% test
# mengimport library random
import random
# melakukan random pada vungsi imgs
random.shuffle(imgs)
```

```
In [30]: # In[1]:import lib
         # menimport libtari CSV untuk mengolah data ber ekstensi csv
         import csv
         #kemudian mengimport librari Image yang berguna untuk dari PIL atau Python Imag
         from PIL import Image as pil_image
         # kemudian mengimport librari keras yang menggunakan method preprocessing yang
         import keras.preprocessing.image
```

Gambar 7.24 hasil kode program

```
# membuat variabel split_idx dengan nilai integer 80 persen dikali dari pengembalian jumlah dari variabel imgs
split_idx = int(0.8*len(imgs))
# membuat variabel train dengan isi lebih besar split idx
train = imgs[:split_idx]
# membuat variabel test dengan isi lebih kecil split idx
test = imgs[split_idx:]
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [32]: import random
         random.shuffle(imgs)
         split_idx = int(0.8*len(imgs))
         train = imgs[:split_idx]
         test = imgs[split_idx:]
```

Gambar 7.25 hasil kode program

4. Jelaskan kode program pada blok In[4].

```
# In[4]:
# mengimport librari numpy dengan inisial np
import numpy as np
# membuat variabel train input dengan np method asarray yang mana membuat array dengan isi row 2 dari data train
train_input = np.asarray(list(map(lambda row: row[2], train)))
# membuat test input input dengan np method asarray yang mana membuat array dengan isi row 2 dari data test
test_input = np.asarray(list(map(lambda row: row[2], test)))
# membuat variabel train_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data train
train_output = np.asarray(list(map(lambda row: row[1], train)))
# membuat variabel test_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data test
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [33]: import numpy as np

         train_input = np.asarray(list(map(lambda row: row[2], train)))
         test_input = np.asarray(list(map(lambda row: row[2], test)))

         train_output = np.asarray(list(map(lambda row: row[1], train)))
         test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Gambar 7.26 hasil kode program

5. Jelaskan kode program pada blok In[5].

```
# In[5]: import encoder and one hot
# mengimport librari LabelEncode dari sklearn
from sklearn.preprocessing import LabelEncoder
# mengimport librari OneHotEncoder dari sklearn
from sklearn.preprocessing import OneHotEncoder
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [34]: from sklearn.preprocessing import LabelEncoder
         from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.27 hasil kode program

6. Jelaskan kode program pada blok In[6].

```
# In[6]:convert class names into one-hot encoding
# membuat variabel label_encoder dengan isi LabelEncoder
label_encoder = LabelEncoder()
# membuat variabel integer_encoded yang berfungsi untuk
    mengkonvert variabel classes kedalam bentuk integer
integer_encoded = label_encoder.fit_transform(classes)
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [35]: # first, convert class names into integers
         label_encoder = LabelEncoder()
         integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.28 hasil kode program

7. Jelaskan kode program pada blok In[7].

```
# In[7]:then convert integers into one-hot encoding
# membuat variabel onehot_encoder dengan isi OneHotEncoder
onehot_encoder = OneHotEncoder(sparse=False)
# mengisi variabel integer_encoded dengan isi integer_encoded
    yang telah di convert pada fungsi sebelumnya
integer_encoded = integer_encoded.reshape(len(integer_encoded),
    1)
# mengkonvert variabel integer_encoded kedalam onehot_encoder
onehot_encoder.fit(integer_encoded)
```

Untuk hasilnya dapat di lihat pada gambar berikut.

8. Jelaskan kode program pada blok In[8].

```
# In[8]:convert train and test output to one-hot
# mengkonvert data train output  mengguanakn variabel
    label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# mengkonvert variabel train_output_int kedalam fungsi
    onehot_encoder
```

```
In [36]: # then convert integers into one-hot encoding
         onehot_encoder = OneHotEncoder(sparse=False)
         integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
         onehot_encoder.fit(integer_encoded)

Out[36]: OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
                       handle_unknown='error', sparse=False)
```

Gambar 7.29 hasil kode program

```
train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
# mengkonvert data test_output mengguanakn variabel label_encoder
    kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# mengkonvert variabel test_output_int kedalam fungsi
    onehot_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
# membuat variabel num_classes dengan isi variabel label_encoder
    dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [37]: # convert train and test output to one-hot
         train_output_int = label_encoder.transform(train_output)
         train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
         test_output_int = label_encoder.transform(test_output)
         test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))

         num_classes = len(label_encoder.classes_)
         print("Number of classes: %d" % num_classes)

         Number of classes: 369
```

Gambar 7.30 hasil kode program

9. Jelaskan kode program pada blok In[9].

```
# In[9]: import sequential
# mengimport librari Sequential dari Keras
from keras.models import Sequential
# mengimport librari Dense, Dropout, Flatten dari Keras
from keras.layers import Dense, Dropout, Flatten
# mengimport librari Conv2D, MaxPooling2D dari Keras
from keras.layers import Conv2D, MaxPooling2D
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [38]: from keras.models import Sequential
         from keras.layers import Dense, Dropout, Flatten
         from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.31 hasil kode program

10. Jelaskan kode program pada blok In[10].

```
# In[10]: desain jaringan
# membuat variabel model dengan isian librari Sequential
model = Sequential()
# variabel model di tambahkan librari Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan dengan librari Conv2D 32bit dengan kernel 3 x 3
model.add(Conv2D(32, (3, 3), activation='relu'))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan librari Flatten
model.add(Flatten())
# variabel model di tambahkan librari Dense dengan fungsi tanh
model.add(Dense(1024, activation='tanh'))
# variabel model di tambahkan librari dropout untuk memangkas data tree sebesar 50 persen
model.add(Dropout(0.5))
# variabel model di tambahkan librari Dense dengan data dari num_classes dan fungsi softmax
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
Model: "sequential_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_7 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 6, 6, 32)          0
_________________________________________________________________
flatten_5 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_9 (Dense)              (None, 1024)              1180672
_________________________________________________________________
dropout_3 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_10 (Dense)             (None, 369)               378225
=================================================================
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.32 hasil kode program

11. Jelaskan kode program pada blok In[11].

```
# In[11]: import sequential
# mengimport librari keras callbacks
import keras.callbacks
# membuat variabel tensorboard dengan isi lib keras
tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
    style')
```

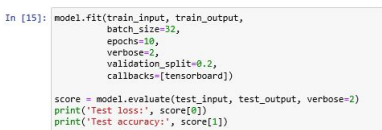
Untuk hasilnya dapat di lihat pada gambar berikut.

Gambar 7.33 hasil kode program

12. Jelaskan kode program pada blok In[12].

```
# In[12]: 5menit kali 10 epoch = 50 menit
# fungsi model titambahkan metod fit untuk mengetahui perhitungan
    dari train_input train_output
model.fit(train_input, train_output,
# dengan batch size 32 bit
          batch_size=32,
          epochs=10,
          verbose=2,
          validation_split=0.2,
          callbacks=[tensorboard])

score = model.evaluate(test_input, test_output, verbose=2)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

Untuk hasilnya dapat di lihat pada gambar berikut.



Gambar 7.34 hasil kode program

13. Jelaskan kode program pada blok In[13].

```
# In[13]:try various model configurations and parameters to find
    the best
# mengimport librari time
import time
#membuat variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
```

```
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # membuat variabel model Sequential
            model = Sequential()
            #membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
            for i in range(conv2d_count):
                # syarat jika i samadengan bobotnya 0
                if i == 0:
                    # menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
                    # jika tidak
                else:
                    # menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                # menambahkan method add pada variabel model dengan isian method  Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
                model.add(MaxPooling2D(pool_size=(2, 2)))
            # merubah feature gambar menjadi 1 dimensi vektor
            model.add(Flatten())
            # menambahkan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
            model.add(Dense(dense_size, activation='tanh'))
            # membuat ketentuan jika pemangkasan lebih besar dari 0 persen
            if dropout > 0.0:
                # menambahkan method dropout pada model dengan nilai dari dropout
                model.add(Dropout(dropout))
                # menambahkan method dense dengan fungsi num classs dan rumus softmax
            model.add(Dense(num_classes, activation='softmax'))
            # mongkompile variabel model dengan hasi loss optimasi dan akurasi matrix
            model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
            # melakukan log pada dir
            log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
            # membuat variabel tensorboard dengan isian dari librari keras dan nilai dari lig dir
            tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
```

```
            # membuat variabel start dengan isian dari librari
    time menggunakan method time

            start = time.time()
            # menambahkan method fit pada model dengan data dari
    train input train output nilai batch nilai epoch verbose
    nilai 20 persen validation split dan callback dengan nilai
    tnsorboard.
            model.fit(train_input, train_output, batch_size=32,
    epochs=10,
                      verbose=0, validation_split=0.2, callbacks
    =[tensorboard])
            # membuat variabel score dengan nilai evaluasi dari
    model menggunakan data tes input dan tes output
            score = model.evaluate(test_input, test_output,
    verbose=2)
            # membuat variabel end
            end = time.time()
            # membuat variabel elapsed
            elapsed = end - start
            # mencetak hasil perhitungan
            print("Conv2D count: %d, Dense size: %d, Dropout: %.2
    f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count
    , dense_size, dropout, score[0], score[1], elapsed))
            results.append((conv2d_count, dense_size, dropout,
    score[0], score[1], elapsed))
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
In [18]: import time

results = []
for conv2d_count in [1, 2]:
    for dense_size in [128, 256, 512, 1024, 2048]:
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            model = Sequential()
            for i in range(conv2d_count):
                if i == 0:
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
                else:
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                model.add(MaxPooling2D(pool_size=(2, 2)))
            model.add(Flatten())
            model.add(Dense(dense_size, activation='tanh'))
            if dropout > 0.0:
                model.add(Dropout(dropout))
            model.add(Dense(num_classes, activation='softmax'))

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

            log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
            tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)

            start = time.time()
            model.fit(train_input, train_output, batch_size=32, epochs=10,
                      verbose=0, validation_split=0.2, callbacks=[tensorboard])
            score = model.evaluate(test_input, test_output, verbose=2)
            end = time.time()
            elapsed = end - start
            print("Conv2D count: %d, Dense size: %d, Dropout: %.2f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count, d
            results.append((conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
```

Gambar 7.35 hasil kode program

14. Jelaskan kode program pada blok In[14].

```
# In[14]:rebuild/retrain a model with the best parameters (from
    the search) and use all data
# membuat variabel model dengan isian librari Sequential
model = Sequential()
```

```
# variabel model di tambahkan librari Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan dengan librari Conv2D 32bit dengan kernel 3 x 3
model.add(Conv2D(32, (3, 3), activation='relu'))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan librari Flatten
model.add(Flatten())
# variabel model di tambahkan librari Dense dengan fungsi tanh
model.add(Dense(128, activation='tanh'))
# variabel model di tambahkan librari dropout untuk memangkas data tree sebesar 50 persen
model.add(Dropout(0.5))
# variabel model di tambahkan librari Dense dengan data dari num_classes dan fungsi softmax
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

Untuk hasilnya dapat di lihat pada gambar berikut.

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 32)          0
_________________________________________________________________
flatten_2 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               147584
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

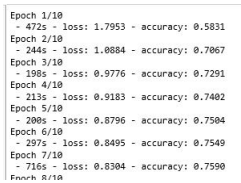
Gambar 7.36 hasil kode program

15. Jelaskan kode program pada blok In[15].

```
# In[15]:join train and test data so we train the network on all data we have available to us
# melakukan join numpy menggunakan data train_input test_input
```

```
model.fit(np.concatenate((train_input, test_input)),
          # kelanjutan data yang di gunakan pada join
    train_output test_output
          np.concatenate((train_output, test_output)),
          #menggunakan ukuran 32 bit dan epoch 10
          batch_size=32, epochs=10, verbose=2)
```

Untuk hasilnya dapat di lihat pada gambar berikut.

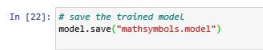


Gambar 7.37 hasil kode program

16. Jelaskan kode program pada blok In[16].

```
# In[16]:save the trained model
#menyimpan model atau mengeksport model yang telah di jalantadi
model.save("mathsymbols.model")
```

Untuk hasilnya dapat di lihat pada gambar berikut.



Gambar 7.38 hasil kode program

17. Jelaskan kode program pada blok In[17].

```
# In[17]:save label encoder (to reverse one-hot encoding)
# menyompan label encoder dengan nama classes.npy
np.save('classes.npy', label_encoder.classes_)
```

Untuk hasilnya dapat di lihat pada gambar berikut.

18. Jelaskan kode program pada blok In[18].

```
# In[18]:load the pre-trained model and predict the math symbol
    for an arbitrary image;
# the code below could be placed in a separate file
# mengimpport librari keras model
import keras.models
# membuat variabel model2 untuk meload model yang telah di simpan
     tadi
model2 = keras.models.load_model("mathsymbols.model")
# mencetak hasil model2
print(model2.summary())
```

```
import keras.models
model2 = keras.models.load_model("mathsymbols.model")
print(model2.summary())
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 6, 6, 32)          0
_________________________________________________________________
flatten_2 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_3 (Dense)              (None, 128)               147584
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.39 hasil kode program

```
In [24]: # save label encoder (to reverse one-hot encoding)
         np.save('classes.npy', label_encoder.classes_)
```

Gambar 7.40 hasil kode program

Untuk hasilnya dapat di lihat pada gambar berikut.

19. Jelaskan kode program pada blok In[19].

```
# In[19]:restore the class name to integer encoder
# membuat variabel label encoder ke 2 dengan isian fungsi label encoder.
label_encoder2 = LabelEncoder()
# menambahkan method classess dengan data classess yang di eksport tadi
label_encoder2.classes_ = np.load('classes.npy')
# membuat fumgsi predict dengan path img
def predict(img_path):
    # membuat variabel newimg dengam membuay immage menjadi array dan membuka data berdasarkan img path
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path))
    # membagi data yang terdapat pada variabel newimg sebanyak 255
    newimg /= 255.0

    # do the prediction
    # membuat variabel predivtion dengan isian variabel model2 menggunakan fungsi predic dengan syarat variabel newimg dengan data reshape
    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

    # figure out which output neuron had the highest score, and reverse the one-hot encoding
    # membuat variabel inverted  denagan label encoder2 dan menggunakan argmax untuk mencari skor luaran tertinggi
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)])
```

```
    # mencetak prediksi gambar dan confidence dari gambar.
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
    max(prediction)))
```

Untuk hasilnya dapat di lihat pada gambar berikut.

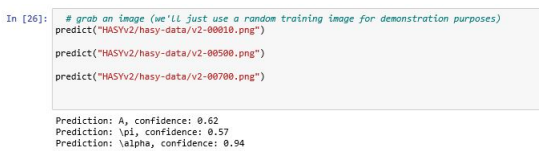
```
In [25]: # restore the class name to integer encoder
label_encoder2 = LabelEncoder()
label_encoder2.classes_ = np.load('classes.npy')

def predict(img_path):
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path))
    newimg /= 255.0

    # do the prediction
    prediction = model2.predict(newimg.reshape(1, 32, 32, 3))

    # figure out which output neuron had the highest score, and reverse the one-hot encoding
    inverted = label_encoder2.inverse_transform([np.argmax(prediction)]) # argmax finds highest-scoring output
    print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

Gambar 7.41 hasil kode program

20. Jelaskan kode program pada blok In[20].

```
# In[20]: grab an image (we'll just use a random training image
    for demonstration purposes)
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi
    dari data di HASYv2/hasy-data/v2-00010.png
predict("HASYv2/hasy-data/v2-00010.png")
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi
    dari data di HASYv2/hasy-data/v2-00500.png
predict("HASYv2/hasy-data/v2-00500.png")
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi
    dari data di HASYv2/hasy-data/v2-00700.png
predict("hasy-data/v2-00700.png")
```

Untuk hasilnya dapat di lihat pada gambar berikut.



```
In [26]: # grab an image (we'll just use a random training image for demonstration purposes)
predict("HASYv2/hasy-data/v2-00010.png")

predict("HASYv2/hasy-data/v2-00500.png")

predict("HASYv2/hasy-data/v2-00700.png")

Prediction: A, confidence: 0.62
Prediction: \pi, confidence: 0.57
Prediction: \alpha, confidence: 0.94
```

Gambar 7.42 hasil kode program