

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 Tunnisia

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. Sariasisih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara
apapun tanpa ijin tertulis dari penerbit

*'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

1 Chapter 1	1
2 Chapter 2	3
3 Chapter 3	5
4 Chapter 4	7
5 Chapter 5	9
6 Chapter 6	21
7 Chapter 7	37

DAFTAR ISI

Foreword	xi
Kata Pengantar	xiii
Acknowledgments	xv
Acronyms	xvii
Glossary	xix
List of Symbols	xxi
Introduction <i>Rolly Maulana Awangga, S.T., M.T.</i>	xxiii
1 Chapter 1	1
2 Chapter 2	3
3 Chapter 3	5
4 Chapter 4	7
	ix

5 Chapter 5	9
5.1 1174002 Habib Abdul Rasyid	9
5.1.1 Teori	9
5.1.2 Praktek	11
6 Chapter 6	21
6.1 Muhammad Tomy Nur Maulidy - 1174031	21
6.1.1 Teori	21
6.1.2 Praktek	25
7 Chapter 7	37
7.1 1174006 - Kadek Diva Krishna Murti	37
7.1.1 Teori	37
7.1.2 Praktek	40
7.1.3 Penanganan Error	48
7.1.4 Bukti Tidak Plagiat	48
7.2 Muhammad Tomy Nur Maulidy 1174031	49
7.2.1 Teori	49
7.2.2 Praktek	55
7.2.3 Penanganan Error	76
Daftar Pustaka	79
Index	81

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

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[1].

$$ABC\mathcal{DEF}\alpha\beta\Gamma\Delta \sum_{def}^{abc} \quad (I.1)$$

BAB 1

CHAPTER 1

BAB 2

CHAPTER 2

BAB 3

CHAPTER 3

BAB 4

CHAPTER 4

BAB 5

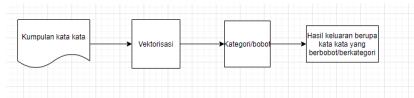
CHAPTER 5

5.1 1174002 Habib Abdul Rasyid

5.1.1 Teori

1. Jelaskan Kenapa Kata-Kata harus dilakukan vektorisasi lengkapi dengan ilustrasi gambar.

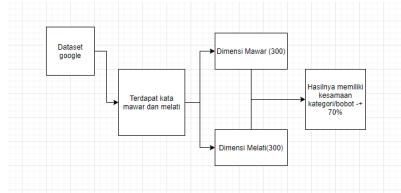
Karena Bertujuan untuk memprediksi kemunculan suatu kata sebuah kalimat atau vektorisasi kata ini juga dapat memprediksikan suatu nilai,bobot, atau kategori yang sama, misalnya ada kategori kendaraan, motor dan mobil termasuk kedalam kategori kendaraan sehingga kita dapat memprediksi kemunculan kendaraan motor dan mobil pada kategori kendaraan.



Gambar 5.1 Teori 1

2. Jelaskan Mengapa dimensi dari vektor dataset google bisa mencapai 300 lengakapi dengan ilustrasi gambar.

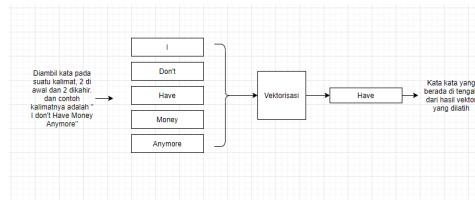
Dimensi dataset dari google bisa mencapai 300 karena dimensi dari vektor tersebut digunakan untuk membandingkan bobot dari setiap kata, misalkan terdapat kata mawar dan melati pada dataset google tersebut setiap kata tersebut dibuat dimensi vektor 300 untuk kata mawar dan 300 dimensi vektor juga untuk kata melati kemudian kata tersebut dibandingkan bobot kesamaan katanya maka akan muncul akurasi sekitar 70 persen kesamaan bobot dikarenakan kata mawar dan melati sama sama di gunakan untuk jenis bunga.



Gambar 5.2 Teori 2

3. Jelaskan Konsep vektorisasi untuk kata . dilengkapi dengan ilustrasi atau gambar.

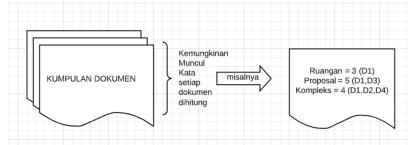
Vektorisasi untuk kata untuk mengetahui kata tengah dari suatu kalimat atau kata utama atau objek utama pada suatu kalimat contoh (Jangan lupa subscribe channel saya ya sekian treimakasih) kata tengah tersebut merupakan channel yang memiliki bobot sebagai kata tengah dari suatu kalimat atau bobot sebagai objek dari suatu kalimat. hal ini sangat berkaitan dengan dimensi vektor pada dataset google yang 300 tadi karena untuk mendapatkan nilai atau bobot dari kata tengah tersebut di dapatkan dari proses dimensiasi dari kata tersebut.



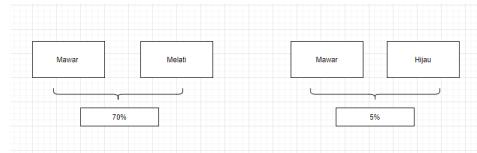
Gambar 5.3 Teori 3

4. Jelaskan Konsep vektorisasi untuk dokumen. dilengkapi dengan ilustrasi atau gambar.

Vektorisasi untuk dokumen hampir sama seperti vektorisasi untuk kata hanya saja pemilihan kata utama atau kata tengah terdapat pada satu dokumen jadi mesin akan membuat dimensi vektor 300 untuk dokumen dan nanti kata tengahnya akan di sandingkan pada dokumen yang terdapat pada dokumen tersebut.

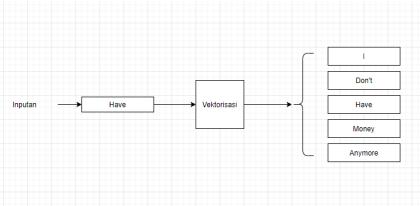
**Gambar 5.4** Teori 4

5. Jelaskan apa mean dan standar deviasi, lengkapi dengan ilustrasi atau gambar.
- mean merupakan petunjuk terhadap kata-kata yang diolah jika kata-kata itu akurasinya tinggi berarti kata tersebut sering muncul begitu juga sebaliknya, sedangkan standar deviation merupakan standar untuk menimbang kesalahan. sehingga kesalahan tersebut dianggap wajar misalkan kita memperkirakan kedekatan dari dataset merupakan 2 atau 3 tapi pada kenyataannya merupakan 5 itu merupakan kesalahan tapi masih bisa dianggap wajar karena masih mendekati perkiraan awal.

**Gambar 5.5** Teori 5

6. Jelaskan Apa itu Skip-Gram sertakan contoh ilustrasi.

Skip-Gram adalah kebalikan dari konsep vektorisasi untuk kata dimana kata tengah menjadi acuan terhadap kata-kata pelengkap dalam suatu kalimat.

**Gambar 5.6** Teori 6

5.1.2 Praktek

1. Mencoba datasets GoogleNews-vectors

- berikut adalah hasil dari code yang digunakan untuk memanggil data library GENSIM dengan menggunakan perintah import, lalu dari library tersebut

diambilah data yang akan digunakan untuk memproses data dari GoogleNews-vector. ilustrasi dapat dilihat pada gambar

```
In [4]: import gensim
In [4]: glove = gensim.models.KeyedVectors.load_word2vec_format('GoogleNews-vectors-negative300.bin', binary=True)
```

Gambar 5.7 import gensim dan olah data GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data LOVE yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [5]: glove['love']
Out[5]:
array([-0.18302734, -0.15334575,  0.40579794,  0.16053905, -0.15533906,
       -0.08242575,  0.26387186,  0.148625,  0.20171269,
      -0.03242432, -0.08283125, -0.02770995, -0.04394531, -0.23535156,
     0.16992188,  0.12890625,  0.15722656,  0.00756835, -0.06982422,
     0.04921888,  0.12890625,  0.15722656,  0.00756835, -0.06982422,
     0.03515625,  0.05537578,  0.10893359,  0.11181641, -0.16388594,
    -0.11181641,  0.13964644,  0.0156396,  0.12792969,  0.15429688,
     0.1865344,  0.12904688,  0.07800876,  0.02600008, -0.10644531,
     0.1865344,  0.12904688,  0.07800876,  0.02600008, -0.10644531,
    -0.18253906,  0.12904688,  0.07800876,  0.02209473,  0.05834961,
```

Gambar 5.8 hasil olah data LOVE pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data FAITH yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [9]: genmod['faith']
Out[9]:
array([-0.26367188, -0.04156391,  0.1953125,  0.13476562, -0.14648458,
       0.11962891,  0.04345793,  0.10551562,  0.12206331,  0.13476562,
      -0.15332803,  0.1883944,  0.18253986, -0.01267188,  0.23144531,
     0.32093125,  0.18449219,  0.36132812, -0.1953125, -0.18164602,
     0.15332803, -0.1883944,  0.18253986, -0.01267188,  0.23144531,
     0.03515625,  0.05537578,  0.10893359,  0.11181641, -0.16388594,
    -0.11181641,  0.13964644,  0.0156396,  0.12792969,  0.15429688,
     0.1865344,  0.12904688,  0.07800876,  0.02600008, -0.10644531,
     0.1865344,  0.12904688,  0.07800876,  0.02600008, -0.10644531,
    -0.1328125,  0.21484375,  0.0135254,  0.02111816,  0.18554688,
     0.3225,  0.06640625, -0.17675781, -0.01780964, -0.1648625,
     0.3225,  0.06640625, -0.17675781, -0.01780964, -0.1648625,
     0.02819824,  0.01257324, -0.09521484, -0.18066406, -0.148625,
```

Gambar 5.9 hasil olah data FAITH pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data FALL yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [10]: genmod['fall']
Out[10]:
array([ 0.04272461,  0.10742188, -0.09277544,  0.16894531, -0.1328125 ,
       -0.18693359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
      -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
     -0.10893359,  0.04321289,  0.01964297,  0.14648458, -0.1589862,
```

Gambar 5.10 hasil olah data FALL pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data SICK yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [11]: gennod['skc']
Out[11]:
1.02617818e-01, 1.4941462e-01, -4.05273438e-02, 1.64052600e-01,
-2.5976525e-01, 3.2226525e-01, 1.73826125e-01, -1.47460931e-01,
1.0174219e-01, 5.4687508e-02, 1.66992188e-01, 9.6485312e-01,
2.24304919e-03, 9.66776572e-02, 1.66991525e-01, 1.12034686e-01,
1.6681563e-01, 1.7887537e-01, 5.92841619e-03, 2.45167238e-01,
8.0734283e-02, 1.7887537e-01, 1.6681563e-01, 4.8886545e-02,
1.77511238e-01, 9.810521238e-02, 1.6681563e-01, 1.7887537e-01,
1.8164602e-01, 2.6562508e-01, -1.45507121e-01, 9.6485312e-01,
9.42382812e-02, 3.12500000e-02, 9.78474509e-02, 6.39544832e-02
```

Gambar 5.11 hasil olah data SICK pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data CLEAR yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [12]: genmod(['clear'])
Out[12]:
array([-2.4416052e-04, -1.02450781e-01, -1.49414062e-01, -4.24804668e-02,
       -1.7698750e-01, -1.46484375e-01, 1.76757812e-01, 1.46484375e-01,
       2.25652500e-01, -2.57582500e-02, -2.57582125e-01, -1.29882125e-01,
       1.24511250e-01, -2.37651250e-01, -2.37651250e-01, -2.37651250e-01,
       2.00458000e-01, -2.00458000e-01, -2.00458000e-01, -2.00458000e-01,
       3.22265200e-02, -3.14431250e-01, -1.11616490e+01, 8.00871250e-01,
       -2.75878906e-06, -6.04248478e-03, -7.37405688e-01, -2.15751526e-01,
       9.66796875e-02, -4.91313088e-01, -8.77109138e-01, -1.40308585e-01,
       7.91815625e-01, -1.79718156e-01, -1.01531562e-01, 8.34963993e-02,
```

Gambar 5.12 hasil olah data CLEAR pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data SHINE yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [13]: genmod['shine']
Out[13]:
array([-0.12402344,  0.39576562, -0.15917969, -0.27734375,  0.30273438,
       0.09969374,  0.39257012, -0.22949219, -0.18359375,  0.3671875,
      -0.10362734,  0.13671875,  0.25396025,  0.17128986,  0.25593602,
      0.21777344,  0.24042348,  0.18375473,  0.13264686,  0.13935938,
      -0.08937749,  0.18022793,  0.08104488,  0.08177344,  0.14648488,
      0.28019572,  0.38080848,  0.08104488,  0.08177344,  0.14648488,
      -0.26755652,  0.03562454,  0.29352781,  0.04516602,  0.08970126,
      -0.24212894,  0.18045112, -0.15234375,  0.05483364,  0.14532612,
      0.396258,   0.2190375,  0.14845735, -0.113183594,  0.14532612,
```

Gambar 5.13 hasil olah data SHINE pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data BAG yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [14]: genmod['bag']  
Out[14]:
```

	-0.03515625	-0.15234375	-0.12402344	0.13378996	-0.11328125
-0.0133667	-0.16113281	0.14648483	-0.06859598	0.140625	
-0.06005859	-0.3046785	0.20996094	0.04345793	-0.2109375	
-0.05957031	-0.05053711	0.10259306	0.19042693	0.04423828	
-0.18847656	-0.07958964	-0.10351556	0.79710156	0.06347565	
-0.18527344	-0.18945532	0.11128122	0.27559062	-0.06771909	
-0.05415925	-0.01810368	0.24121094	-0.18785552	0.05668559	
-0.09514453	-0.16113286	0.16599636	-0.09571454	0.15668559	

Gambar 5.14 hasil olah data BAG pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data CAR yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [15]: genmod ['car']
Out[15]:
array([ 9.1388939e-01,  0.0084228e-01,  -0.0344727e-01,  -0.04883379e-01,
       0.14257812e-01,  0.04931641e-01,  -0.16894531e-01,  -0.20886485e-01,
       0.18866406e-01,  -0.25e-01,  -0.18400391e-01,  -0.10742188e-01,
       0.11882110e-01,  -0.11400391e-01,  -0.10742188e-01,  -0.10742188e-01,
       0.16113281e-01,  -0.01511128e-01,  -0.03988379e-01,  -0.08447266e-01,
       0.16213935e-01,  -0.04467773e-01,  -0.15527344e-01,  -0.25986252e-01,
       0.33964375e-01,  -0.89758636e-01,  -0.16593996e-01,  -0.06847466e-01,
       0.16593996e-01,  -0.18945312e-01,  -0.02832031e-01,
       0.05346688e-01,  -0.03983965e-01,  -0.11883984e-01,  -0.24121094e-01,
       -0.234375e-01,  -0.234375e-01,  -0.234375e-01,  -0.234375e-01,  -0.234375e-01,
```

Gambar 5.15 hasil olah data CAR pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data WASH yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [16]: genmod ['wash']
Out[16]:
array([ 9.46044922e-01,  1.41091502e-01,  -5.46875000e-02,  1.34798625e-01,
       0.35221258e-01,  3.24218750e-01,  -0.44726562e-02,  -1.29802812e-01,
       0.17910156e-01,  2.53906250e-01,  1.13525391e-02,  -1.66992188e-01,
       0.21375000e-01,  -0.15527344e-01,  -0.25986252e-01,  -0.33964375e-01,
       0.21194802e-01,  -0.04679500e-01,  -2.11948022e-01,  -0.88671875e-02,
       0.42137500e-02,  2.12898025e-01,  1.74508476e-02,  2.02941895e-03,
       0.11882110e-01,  -0.11400391e-01,  -0.10742188e-01,  -0.10742188e-01,
       -0.67628899e-03,  -0.13885938e-02,  -2.38281259e-01,  2.38632812e-01,
```

Gambar 5.16 hasil olah data WASH pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data MOTOR yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [17]: genmod ['motor']
Out[17]:
array([ 1.73738469e-02,  1.58398625e-01,  -4.61425781e-02,  -1.32812500e-01,
       0.34779625e-01,  -1.18263719e-01,  -0.74804688e-02,  -2.44160252e-01,
       0.57823625e-01,  -0.15527344e-01,  -0.25986252e-01,  -0.33964375e-01,
       0.21194802e-01,  -0.04679500e-01,  -2.11948022e-01,  -0.88671875e-02,
       0.51953125e-01,  5.76171875e-02,  8.15429688e-02,  1.86765798e-02,
       0.21375000e-01,  -0.15527344e-01,  -0.25986252e-01,  -0.33964375e-01,
       0.15625000e-01,  -0.42894646e-02,  -1.32812500e-01,  2.11948022e-01,
       0.23846750e-01,  1.69921875e-01,  -1.55273438e-01,  4.39363750e-01,
       0.11882110e-01,  -0.11400391e-01,  -0.10742188e-01,  -0.10742188e-01,
       -0.26538866e-03,  2.28515625e-01,  8.84837500e-02,  -7.12896252e-02,
```

Gambar 5.17 hasil olah data MOTOR pada GoogleNews-vector

- lalu pada penggunaan code berikut ini akan mengolah data CYCLE yang terdapat pada file GoogleNews-vector, hasil dari pemrosesannya dapat dilihat pada gambar

```
In [18]: genmod ['cycle']
Out[18]:
array([ 0.04541016,  0.21579688,  -0.02799061,  -0.10333512,
       0.18263719,  -0.15829312,  -0.06177675,  0.12172656,  -0.15829312,
       0.02563477,  -0.07568359,  -0.0625,  -0.04614245,  -0.11054688,
       0.07226562,  -0.06445312,  0.05517578,  0.14941406,  0.13671875,
       0.16902734,  0.02172652,  -0.10693359,  0.02409244,  -0.10644531,
       0.02563477,  -0.07568359,  -0.0625,  -0.04614245,  -0.11054688,
       0.17873994,  0.01165771,  -0.01696777,  0.13671875,  -0.16460252,
```

Gambar 5.18 hasil olah data CYCLE pada GoogleNews-vector

- dan pada hasil code berikut ini adalah hasil dari proses penggunaan perintah code similarity yang akan menghitung nilai value data yang dibandingkan dengan masing - masing kata seperti pada hasil dari perbandingan kata LOVE disandingkan dengan FAITH menghasilkan nilai 37 persen, sedangkan kata WASH dan SHINE menghasilkan nilai 27 persen dan kata CAR yang disandingkan dengan kata MOTOR menghasilkan 48 persen, dimana kita dapat menyimpulkan bahwa semakin data kata tersebut memiliki

tingkat kesamaan yang tinggi maka nilai hasil yang ditampilkan akan semakin tinggi. ilustrasi bisa dilihat pada gambar

```
In [19]: genmod.similarity('love', 'faith')
Out[19]: 0.3705347934587281

In [20]: genmod.similarity('wash', 'shine')
Out[20]: 0.2770128965426825

In [21]: genmod.similarity('car', 'motor')
Out[21]: 0.4810172852001571

In [22]: genmod.similarity('bag', 'cycle')
Out[22]: 0.040672699213443584

In [23]: genmod.similarity('shine', 'fall')
Out[23]: 0.27789493775772145
```

Gambar 5.19 hasil olah data pada GoogleNews-vector menggunakan SIMILARITY

2. extract_words dan PermutatedSentences

pada penjelasan berikut ini akan menyangkut pembersihan data yang akan digunakan untuk diproses, dimana data akan di EXTRACT dari setiap katanya agar terbebas dari data TAG HTML, APOSTROPES, TANDA BACA, dan SPASI yang berlebih. dengan menggunakan perintah code STRIP dan SPLIT. lalu penggunaan library random yang akan dibuat untuk melakukan KOCLOK data dengan acuan datanya adalah data yang terdapat pada variable KATA. untuk ilustrasi hasil dari codenya dapat dilihat pada gambar

```
In [28]: import re
...: def extract_words(kata):
...:     kata = kata.lower()
...:     kata = re.sub("<[^>*>]", " ", kata)
...:     kata = re.sub("\n", "\n\n", kata)
...:     kata = re.sub("\r", "\r\r", kata)
...:     kata = re.sub("\t", "\t\t", kata)
...:     kata = re.sub("\s+", " ", kata)
...:     kata = kata.strip()
...:     return kata.split()

...: import random
...: class PermutatedSentences(object):
...:     def __init__(self, lenght):
...:         self.lenght = lenght
...:     def __iter__(self):
...:         req = list(self.lenght)
...:         random.shuffle(req)
...:         for kata in req:
...:             yield kata
```

Gambar 5.20 hasil olah data pada GoogleNews-vector menggunakan extract_words dan PermuteSentences

3. TaggedDocument dan Doc2Vec

gensim merupakan open-source model ruang vektor dan toolkit topic modeling, yang diimplementasikan dalam bahasa pemrograman Python. Untuk kerja Gensim, digunakan NumPy, SciPy dan Cython (opsional). Gensim secara khusus ditujukan untuk menangani koleksi teks besar dengan menggunakan algoritma secara online. Gensim mengimplementasikan tf-idf, latent semantic analysis (LSA), Latent Dirichlet Analysis (LDA), dan lain-lain.

tagged document merupakan sebuah class yang terdapat pada pemrosesan data pada library gensim yang akan mengolah data teks yang ada pada dokumen - dokumen yang dipakai.

Doc2Vec merupakan algoritma doc embedding, yaitu pemetaan dari dokumen menjadi vektor, serta pemetaan data dokumen 1 dan dokumen lainnya. Ilustrasi dari tagged document dan Word2Vec ada pada gambar

```
In [2]: from gensim.models.doc2vec import TaggedDocument  
...: from gensim.models import Doc2Vec
```

Gambar 5.21 TaggedDocument dan Doc2Vec

4. Praktek data training

pertama buka data training yang akan diolah pada aplikasi python, import library OS dan membuat data variable unsup_sentences dengan nilai array kosong. buatkan data direktori untuk memanggil data yang akan diolah dan buatkan juga variable data nilai fname yang akan memproses data dirname untuk diisikan pada variable unsup_sentences. code yang digunakan dapat dilihat pada gambar

```

    untagged_sentences = []
    tagged_sentences = []

    for sentence in sentences:
        tokens = sentence['tokens']
        pos_tags = sentence['pos_tags']

        if len(tokens) > 1:
            for i in range(1, len(tokens)):
                if tokens[i] == 'I' and tokens[i-1] == 'B':
                    if pos_tags[i] == 'NNP' or pos_tags[i] == 'NNPS':
                        untagged_sentence = sentence['text'].replace(tokens[i], 'I')
                        tagged_sentence = sentence
                    else:
                        untagged_sentence = sentence['text'].replace(tokens[i], 'I')
                        tagged_sentence = sentence
                else:
                    untagged_sentence = sentence['text']
                    tagged_sentence = sentence

                untagged_sentences.append(untagged_sentence)
                tagged_sentences.append(tagged_sentence)

    return untagged_sentences, tagged_sentences

```

Gambar 5.22 data code praktik data training

data pada hasil code digambar berikut 5.23, menghasilkan data pada gambar 5.24 yang akan memunculkan data variable DIRNAME, FNAME, KATA dan unsup_sentences yang memiliki data sebanyak 55 kata dalam file yang diolah tersebut. hasil run dengan menggunakan code pada gambar 5.25, menghasilkan data nilai yang terdapat pada gambar 5.26. lalu pada code yang terdapat digambar 5.27, menghasilkan data 5.28.

```
[102]:  
import os  
unsup_sentences = []  
  
for dirname in ["train/pos", "train/neg", "train/unsup", "test/pos", "test/neg"]:  
    for fname in sorted(os.listdir(dirname + "/")):  
        if fname[-4:] == ".txt":  
            with open(dirname + "/" + fname, encoding='UTF-8') as f:  
                lines = f.readlines()  
                words = extract_words(lines)  
            unsup_sentences.append(TaggedDocument(words, [dirname + "/" + fname]))
```

Gambar 5.23 data code praktik data training

Gambar 5.24 data code praktik data training

```
for dirname in ['review_polarity/txt_sentoken/pos', 'review_polarity/txt_sentoken/neg']:
    for frame in sorted(os.listdir(dirname)):
        if frame.endswith('.tsv'):
            with open(dirname + '/' + frame, encoding='UTF-8') as f:
                for i, kata in enumerate(f):
                    if i > 0:
                        words = kata.split()
                        append_sentences.append(createDocument(words, ['%s-%s-%d' % (dirname, frame, i)]))
```

Gambar 5.25 data code praktik data training

Name	Type	Size	Value
dname	str	1	review_polarity/txt_sentoken/neg
fname	str	1	c99bf_144630.txt
i	int	1	26
kata	str	1	after watching „A_night_at_the Roxbury„, You'll be left with mainly ...
unsup_sentences	list	130812	[TaggedDocument, TaggedDocument, TaggedDocument, TaggedDocument, Tagge ...
words	list	3	['.', ',', '']

Gambar 5.26 data code praktik data training

```
# In[23]:  
with open('stanfordSentimentTreebank/original_rt_snippets.txt', encoding='UTF-8') as f:  
    for i, line in enumerate(f):  
        words = extract_words(line)  
        unsup_sentences.append(TaggedDocument(words, ["rt-%d" % i]))
```

Gambar 5.27 data code praktik data training

Name	Type	Size	Value
dirname	str	1	review_polarity/txt_sentoken/neg
Frame	str	1	cv999_14636.txt
i	int	1	10004
kata	str	1	after watching _a_night_at_the Roxbury_ , you'll feel like you're walking around in a dream.
line	str	1	Hera fans walked out muttering words like "horrible" and "terrible..."
words_sentences	list	144861	[TaggedDocument, TaggedDocument, TaggedDocument]
words	list	3	['.', ',', '']

Gambar 5.28 data code praktik data training

5. Why need Shuffled and Clean memory

dilakukan shuffled adalah agar datanya lebih mudah untuk diolah dan untuk menentukan tingkat tinggi akurasi dari hasil pemrosesan. dan dilakukan pembersihan memory adalah agar chance yang disimpan tidak membuat proses pada komputer menjadi lambat dan dapat digunakan untuk memproses data lainnya agar menjadi lebih ringan dan cepat. pada gambar 5.29 adalah proses untuk melakukan pengoclokan data dan pada gambar 5.30 adalah proses untuk memasukan data unsup_sentences kedalam variable muter untuk diproses dengan class PermuterSentences. dan pada gambar 5.31 adalah code yang digunakan untuk membersihkan data memory.

```
In [28]: import re
...: def extract_words(kata):
...:     kata = kata.lower()
...:     kata = re.sub("<[^>]+>|' ',' ',kata")
...:     kata = re.sub("(\\w')|('w')|('l')|('2',kata)
...:     kata = re.sub("'\w+',' ',kata)
...:     kata = re.sub("\w+",' ',kata)
...:     kata = kata.strip()
...:     return kata.split()
...:
...:
...: import random
...: class PermuteSentences(object):
...:     def __init__(self, lenght):
...:         self.lenght = lenght
...:
...:     def __iter__(self):
...:         req = list(self.lenght)
...:         random.shuffle(req)
...:         for kata in req:
...:             yield kata
```

Gambar 5.29 Shuffled dan Randomisasi data

```
In [18]: muter = PermuteSentences(unsup_sentences)
...: mod = Doc2Vec(muter, dm=0, hs=1, size=50)
```

Gambar 5.30 pembuatan variable muter untuk memuat data unsup_sentences

```
mod.delete_temporary_training_data(keep_inference=True)
```

Gambar 5.31 code untuk membersihkan data memory

6. Why model have to be saved

dalam pengolahan data dengan menggunakan proses yang panjang ditakutkan data yang sudah diproses tersebut dapat hilang jika terdapat kejadian atau emergency pada saat pengolahan dan pemrosesan data, misalnya harddisk error atau pun listrik yang padam. dan proses penyimpanan data juga dilakukan agar data yang sudah diolah data dipanggil lagi tanpa harus melakukan proses dari awal sehingga tidak memakan waktu. untuk code yang digunakan dapat dilihat pada gambar berikut ini adalah hasil file dari penggunaan code save tersebut. bisa dilihat pada gambar

```
In [13]: mod.infer_vector(extract_words("This Place is not worth your time,
it's a waste of time"))
Out[13]:
array([-0.00774486, -0.0017115, -0.00647213, -0.00179012, 0.00060511,
-0.0058244, 0.00037378, -0.00000683, 0.00724112, 0.00209949,
-0.0037455, -0.0072116, -0.00252699, 0.00921614, 0.00114124,
0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001,
0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001,
0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001,
-0.00567061, 0.00646779, -0.0007159, -0.00392188, 0.0039201,
-0.00054506, -0.00203164, 0.001266, -0.00929867, 0.00763593,
0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001, 0.00000001,
0.00000001, -0.00279454, -0.00753236, 0.000404279, -0.004042311],
dtype='float32')
```

Gambar 5.32 hasil file simpan

7. infer_vector

berfungsi untuk dokumen baru, dan bisa menggunakan data vektor yang dilatih secara massal, seperti yang disimpan dalam model, untuk dokumen yang merupakan bagian dari data training. untuk percobaannya dapat dilihat pada gambar

```
In [12]: from sklearn.metrics.pairwise import cosine_similarity
...: ...
...: mod.infer_vector(extract_words("highly recommended.")),
...: mod.infer_vector(extract_words("service sucks.")))
Out[12]: array([0.208962], dtype=float32)
```

Gambar 5.33 code dan hasil infer_vector

8. cosine_similarity

merupakan sebuah algoritma yang digunakan untuk membandingkan dari dua buah data yang bukan merupakan data vector untuk menguji nilai kemiripan data satu dengan data lainnya. hasil dari percobaan pada tugas no 8 ini dapat dilihat pada gambar 5.34 yang menghasilkan nilai akurasi sebesar 20 persen dan gambar 5.35 yang menghasilkan nilai akurasi sebesar 91 persen.

```
In [13]: from sklearn.metrics.pairwise import cosine_similarity
...: ...
...: mod.infer_vector(extract_words("tolong bantuan.")),
...: mod.infer_vector(extract_words("tolong bantuan.")))
Out[13]: array([0.91143525], dtype=float32)
```

Gambar 5.34 code dan hasil penggunaan cosine_similarity

```
In [16]: from sklearn.neighbors import KNeighborsClassifier
...: from sklearn.ensemble import RandomForestClassifier
...: from sklearn.model_selection import cross_val_score
...: import numpy as np
...: ...
...: clf = KNeighborsClassifier(n_neighbors=9)
...: clfrf = RandomForestClassifier()
```

Gambar 5.35 code dan hasil penggunaan cosine_similarity

9. Cross Validation

pertama melakukan import data dari library KNeighborsClassifier, RandomForestClassifier, cross_val_score dan numpy yang digunakan untuk membuat data cross validasi dapat dilihat pada gambar

```
In [40]: scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
...: np.mean(scores), np.std(scores)
Out[40]: (0.5283333333333334, 0.086411794687223791)
```

Gambar 5.36 memasukan code import library

lalu selanjutnya membuat data variable scores yang akan memuat nilai cross_val_score dengan datanya diambil dari KNeighborsClassifier yang terdiri dari sentvecs, sentiments dan clf dan mengolahnya menggunakan numpy untuk menampilkan data pada gambar yang menghasilkan nilai akurasi sebesar 53 persen.

```
In [41]: scores2 = cross_val_score(clfrf, sentvecs, sentiments, cv=5)
... np.mean(scores2), np.std(scores2))
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
Out[41]: (0.5119999999999999, 0.40644635968604594)
```

Gambar 5.37 perhitungan data KNeighborsClassifier dengan cross validasi

membuat data variable scores yang akan memuat nilai cross_val_score dengan datanya diambil dari RandomForestClassifier yang terdiri dari sentvecs, sentiments dan clfrf dan mengolahnya menggunakan numpy untuk menampilkan data pada gambar yang menghasilkan nilai akurasi sebesar 53 persen.

```
In [42]: from sklearn.pipeline import make_pipeline
... from sklearn.feature_extraction.text import CountVectorizer,
TfidfTransformer
... pipeline = make_pipeline(CountVectorizer(), TfidfTransformer(),
RandomForestClassifier())
... np.mean(scores), np.std(scores))
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Path\PC\Anconda\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in version 0.20
to 100 in 0.22. "In version 0.20 to 100 in 0.22.", FutureWarning)
Out[42]: (0.7416666666666667, 0.8234520787991171)
```

Gambar 5.38 perhitungan data RandomForestClassifier dengan cross validasi

penggunaan make_pipeline adalah untuk membuat data dari KNeighborsClassifier, RandomForestClassifier dan Vectorizer digabungkan untuk menghasilkan data nilai pada gambar menghasilkan nilai akurasi sebesar 74 persen.

scores	float64 (5,)	(0.53333333 0.535 0.52333333 0.51833333
		0.53166667 0.53166667 0.52333333 0.51833333 0.51833333
score2	float64 (5,)	[0.53333333 0.52333333 0.51666667 0.51166667 0.505
		54 0.74666667 0.73333333 0.71833333 0.71166667 0.70533333]
score3	float64 (5,)	[0.53333333 0.52333333 0.51666667 0.51166667 0.505
		54 0.74666667 0.73333333 0.71833333 0.71166667 0.70533333]

Gambar 5.39 perhitungan data Cross Validasi untuk nilai keseluruhan dari KNeighborsClassifier, RandomForestClassifier dan Vectorizer

BAB 6

CHAPTER 6

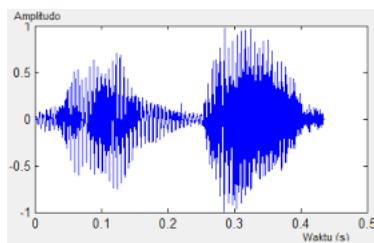
6.1 Muhammad Tomy Nur Maulidy - 1174031

6.1.1 Teori

1. Kenapa file suara harus di lakukan MFCC. dilengkapi dengan ilustrasi atau gambar.

Nilai-nilai MFCC meniru pendengaran manusia dan mereka biasanya digunakan dalam aplikasi pengenalan suara serta genre musik deteksi. Nilai-nilai MFCC ini akan dimasukkan langsung ke jaringan saraf. Agar dapat diubah menjadi bentuk vektor, dan dapat digunakan pada machine learning. Disebabkan machine learning hanya mengerti bilangan vektor saja.

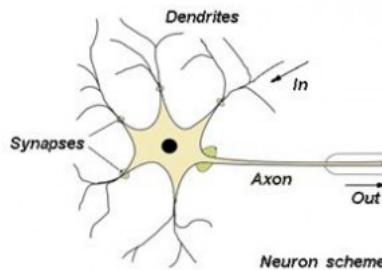
Ilustrasinya, Ketika ingin menggunakan file suara dalam machine learning, misalnya untuk melihat jam. Machine learning tidak memahami rekaman suara melainkan vektor. Maka rekaman tersebut akan diubah kedalam bentuk vektor kemudian vektor akan menyesuaikan dengan kata kata yang sudah disediakan. Jika cocok maka akan mengembalikan waktu yang diinginkan



Gambar 6.1 Contoh MFCC

2. Konsep dasar neural network dilengkapi dengan ilustrasi atau gambar

Neural Network ini terinspirasi dari jaringan saraf otak manusia. Dimana setiap neuron terhubung ke setiap neuron di lapisan berikutnya. Lapisan pertama menerima input dan lapisan terakhir memberikan keluaran. Struktur jaringan, yang berarti jumlah neuron dan koneksinya, diputuskan sebelumnya dan tidak dapat berubah, setidaknya tidak selama training. Juga, setiap input harus memiliki jumlah nilai yang sama. Ini berarti bahwa gambar, misalnya, mungkin perlu diubah ukurannya agar sesuai dengan jumlah neuron input.

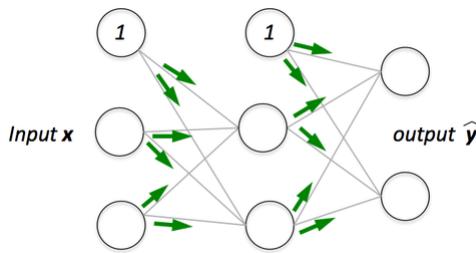


Gambar 6.2 Contoh Pembobotan Neural Network

3. Konsep pembobotan dalam neural network.dilengkapi dengan ilustrasi atau gambar

Bobot mewakili kekuatan koneksi antar unit. Jika bobot dari node 1 ke node 2 memiliki besaran lebih besar, itu berarti bahwa neuron 1 memiliki pengaruh lebih besar terhadap neuron 2. Bobot penting untuk nilai input. Bobot mendekati nol berarti mengubah input ini tidak akan mengubah output. Bobot negatif berarti meningkatkan input ini akan mengurangi output. Bobot menentukan seberapa besar pengaruh input terhadap output. Seperti contoh berikut :

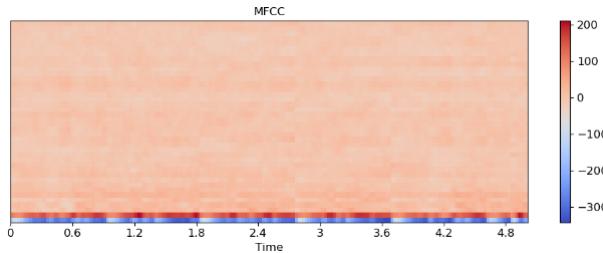
4. Konsep fungsi aktifasi dalam neural network. dilengkapi dengan ilustrasi atau gambar



Gambar 6.3 Contoh Pembobotan Neural Network

Fungsi aktivasi digunakan untuk memperkenalkan non-linearitas ke jaringan saraf. Ini menekan nilai dalam rentang yang lebih kecil yaitu. fungsi aktivasi Sigmoid memeras nilai antara rentang 0 hingga 1. Ada banyak fungsi aktivasi yang digunakan dalam industri pembelajaran yang dalam dan ReLU, SeLU dan TanH lebih disukai daripada fungsi aktivasi sigmoid. Ilustrasinya, ketika fungsi aktivasi linier, jaringan saraf dua lapis mampu mendekati hampir semua fungsi. Namun, jika fungsi aktivasi identik dengan fungsi aktivasi $F(X) = X$, properti ini tidak puas, dan jika MLP menggunakan fungsi aktivasi yang sama, seluruh jaringan setara dengan jaringan saraf lapis tunggal.

5. Cara membaca hasil plot dari MFCC,dilengkapi dengan ilustrasi atau gambar Berikut merupakan hasil plot dari rekaman suara : Dari gambar tersebut dapat



Gambar 6.4 Cara Membaca Hasil Plot MFCC

diketahui :

- Terdapat 2 dimensi yaitu x sebagai waktu, dan y sebagai power atau desibel.
- Dapat dilihat bahwa jika berwarna biru maka power dari suara tersebut rendah, dan jika merah power dari suara tersebut tinggi
- Dibagian atas terdapat warna merah pudar yang menandakan bahwa tidak ada suara sama sekali dalam jangkauan tersebut.

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

One-hot encoding adalah representasi variabel kategorikal sebagai vektor biner. Mengharuskan nilai kategorikal dipetakan ke nilai integer. Kemudian, setiap nilai integer direpresentasikan sebagai vektor biner yang semuanya bernilai nol kecuali indeks integer, yang ditandai dengan 1.

Label Encoding			One Hot Encoding			
Food Name	Categorical #	Calories	Apple	Chicken	Broccoli	Calories
Apple	1	95	1	0	0	95
Chicken	2	231	0	1	0	231
Broccoli	3	50	0	0	1	50

Gambar 6.5 One Hot Encoding

7. fungsi dari np.unique dan to categorical dalam kode program,dilengkapi dengan ilustrasi atau gambar.

Untuk np.unique fungsinya yaitu menemukan elemen unik array. Mengembalikan elemen unik array yang diurutkan. Ada tiga output opsional selain elemen unik:

- Indeks array input yang memberikan nilai unik
- Indeks array unik yang merekonstruksi array input
- Berapa kali setiap nilai unik muncul dalam array input.

```
>>> 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 6.6 Numpy Unique

Untuk To Categorical fungsinya untuk mengubah vektor kelas (integer) ke matriks kelas biner.

```
# 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)
```

Gambar 6.7 To Categorical

8. Fungsi dari Sequential dalam kode program,dilengkapi dengan ilustrasi atau gambar.

Sequential berfungsi sebagai tumpukan linear lapisan. COntohnya sebagai berikut :

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(2, input_dim=1))
model.add(Dense(1))
```

Gambar 6.8 Sequential

6.1.2 Praktek

- Penjelasan isi data GTZAN Genre Collection dan data dari Freesound. Isi data data merupakan datasets lagu atau suara yang terdiri dari 10 genre yang di simpan kedalam 10 folder yaitu folder blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock ke sepuluh folder. Masing - masing dari folder terdapat 100 data suara sedangkan data freesound merupakan contoh data suara yang akan di gunakan untuk menguji hasil pengolahan data tersebut dengan menggunakan metode mfcc. Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:21:58 2020
4
5 @author: User
6 """
7
8 import librosa
9 import librosa.feature
10 import librosa.display
11 import glob
12 import numpy as np
13 import matplotlib.pyplot as plt
14 from keras.layers import Dense, Activation
15 from keras.models import Sequential
16 from keras.utils import to_categorical
```

- Penjelasan perbaris kode program dari display MFCC.

- Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:22:17 2020
4
5 @author: User
6 """
7
```

```

8 def display_mfcc(song):
9     y, _ = librosa.load(song)
10    mfcc = librosa.feature.mfcc(y)
11
12    plt.figure(figsize=(10, 4))
13    librosa.display.specshow(mfcc, x_axis='time',
14                             y_axis='mel')
15    plt.colorbar()
16    plt.title(song)
17    plt.tight_layout()
18    plt.show()

```

▪ Penjelasan:

- Baris Code 1: Membuat fungsi display MFCC untuk menampilkan vektorisasi dari sebuah suara dimana variabel parameter.
- Baris Code 2: Membuat variabel Y dimana untuk membaca variable parameter song dari perintah librosa load.
- Baris Code 3: Membuat variabel MFCC untuk memanggil variabel Y dan mengubah suara menjadi vektor.
- Baris Code 4: Melakukan plotting gambar dengan ukuran 10x4 dari figsize.
- Baris Code 5: Menampilkan spektrogram dari library librosa dimana untuk x_axis didefinisikan dengan time kemudian y_axis di definisikan dengan mel.
- Baris Code 6: Menambahkan colorbar pada plot yang dijalankan.
- Baris Code 7: Menetapkan atau memberikan judul untuk suara yang dieksekusi.
- Baris Code 8: Untuk menyesuaikan subplot params sehingga subplot cocok dengan area gambar.
- Baris Code 9: Fungsi untuk menampilkan hasil plot dari inputan yang telah dieksekusi.

▪ Ilustrasi Gambar:

3. Penjelasan perbaris code dari Extract Feature Song.

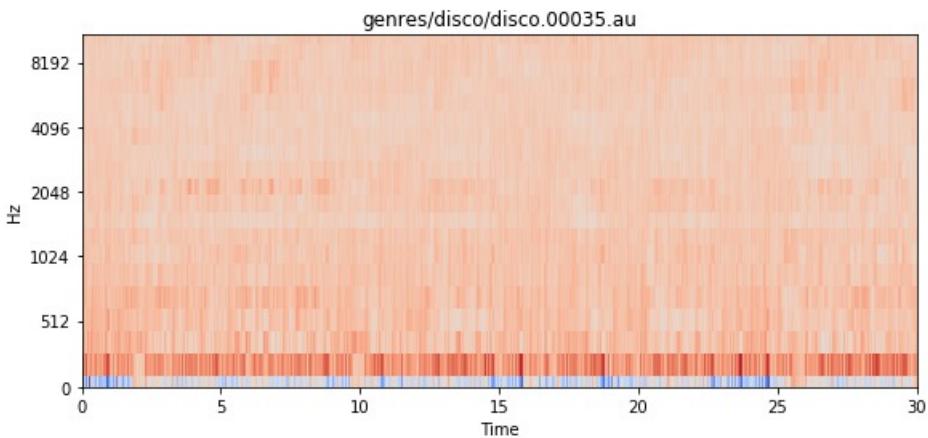
▪ Code yang digunakan:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:25:02 2020
4
5 @author: User
6 """
7
8 def extract_features_song(f):
9     y, _ = librosa.load(f)
10
11    # get Mel-frequency cepstral coefficients
12    mfcc = librosa.feature.mfcc(y)

```

```
In [5]: display_mfcc('genres/disco/disco.00035.au')
```



Gambar 6.9 Display MFCC

```
13 # normalize values between -1,1 (divide by max)
14 mfcc /= np.amax(np.absolute(mfcc))
15
16 return np.ndarray.flatten(mfcc)[:25000]
```

- Penjelasan Code:

- Baris Code 1 : Membuat fungsi extract feature song dengan inputan parameter f
- Baris Code 2 : Membuat variabel y dimana untuk meload atau membaca inputan parameter f dari perintah librosa load song
- Baris Code 3 : Membuat variabel mfcc yang difungsikan untuk membuat feature dari variabel y berdasarkan library librosa
- Baris Code 4 : Membuat normalisasi nilai antara -1 sampai 1 yang didapatkan dari eksekusi np.absolute
- Baris Code 5 : Didefinisikan untuk mengambil 25000 data pertama berdasarkan durasi suara atau musik lalu dikembalikan salinan arraynya dan dikecilkan menjadi satu.

- Ilustrasi Gambar:

- Mengapa yang diambil merupakan 25.000 baris data pertama?
Biar supaya tidak terjadi overhead pada komputer atau laptop atau proses eksekusi tidak terlalu lama.

4. Penjelasan perbaris code dari Generate Features and Labels.

Code yang digunakan:

```
In [57]: 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]
```

Gambar 6.10 Extract Features Song

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:25:14 2020
4
5 @author: User
6 """
7
8 def generate_features_and_labels():
9     all_features = []
10    all_labels = []
11
12    genres = ['blues', 'classical', 'country', 'disco', 'hiphop', 'jazz', 'metal', 'pop', 'reggae', 'rock']
13    for genre in genres:
14        sound_files = glob.glob('genres/' + genre + '/*.au')
15        print('Processing %d songs in %s genre ... ' % (len(sound_files), genre))
16        for f in sound_files:
17            features = extract_features_song(f)
18            all_features.append(features)
19            all_labels.append(genre)
20
21    # convert labels to one-hot encoding cth blues :
22    # 1000000000 classic 0100000000
23    label_uniq_ids, label_row_ids = np.unique(all_labels, return_inverse=True) #ke integer
24    label_row_ids = label_row_ids.astype(np.int32, copy=False)
25    onehot_labels = to_categorical(label_row_ids, len(label_uniq_ids)) #ke one hot
26
27    return np.stack(all_features), onehot_labels
```

▪ Penjelasan:

- Baris Code 1: Membuat perintah untuk fungsi generate features and labels.
- Baris Code 2: Pembuatan variabel all features dengan array atau parameter kosong.
- Baris Code 3: Pembuatan variabel all labels dengan array atau parameter kosong.
- Baris Code 4: Mendefinisikan variable genres yang didalamnya berisi nama folder-folder pada variabel genres tersebut.
- Baris Code 5: Membuat perintah fungsi looping.

- (f) Baris Code 6: Membuat atribut sound files yang berisi perintah looping perfolder dari folder genres dan mengambil semua file berekstensi au.
- (g) Baris Code 7: Memunculkan jumlah song yang dieksekusi.
- (h) Baris Code 8: Membuat perintah fungsi dari sound files.
- (i) Baris Code 9: Membuat variabel features untuk memanggil fungsi extract features song (f) sebagai inputan. Setiap satu file array sound files dilakukan ekstrak fitur.
- (j) Baris Code 10: Memasukkan semua features menggunakan perintah append kedalam all features.
- (k) Baris Code 11: Memasukkan semua genres menggunakan perintah append ke dalam all labels.
- (l) Baris Code 12: Mendefinisikan label uniq ids dan label row ids sebagai variabel dimana mengeksekusi perintah np.unique dengan parameter variabelnya all labels dan return inverse=True.
- (m) Baris Code 13: Membuat variabel label row ids untuk menentukan type dari variabel tersebut dengan type bit yang sesuai dengan yang digunakan.
- (n) Baris Code 14: Membuat variabel onehot labels dimana mengeksekusi to_categorical dengan variabel parameter low row ids dan len.
- (o) Baris Code 15: Mengembalikan dan menampilkan hasil eksekusi dari variabel parameter all features dan onehot labels perintah dari np.stack.

▪ Ilustrasi Gambar:

```
In [58]: 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 classical
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
```

Gambar 6.11 Generate Features and Label

5. Penjelasan penggunaan fungsi Generate Features and Labels sangat lama ketika Meload Dataset Genre.

- Code yang digunakan:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:26:15 2020
4
5 @author: User
6 """
7
8 features , labels = generate_features_and_labels()

```

- Penjelasan:

Baris 1: Variabel features and label akan mengeksekusi isi dari features and label

Baris 2: Memproses 100 lagu di genre blues

Baris 3: Memproses 100 lagu di genre classical

Baris 4: Memproses 100 lagu di genre country

Baris 5: Memproses 100 lagu di genre disco

Baris 6: Memproses 100 lagu di genre hip hop

Baris 7: Memproses 100 lagu di genre jazz

Baris 8: Memproses 100 lagu di genre metal

Baris 9: Memproses 100 lagu di genre pop

Baris 10: Memproses 100 lagu di genre reggae

Baris 11: Memproses 100 lagu di genre rock

- Ilustrasi Gambar:

```

Processing 100 songs in blues genre...
Processing 100 songs in classical genre...
Processing 100 songs in country genre...
Processing 100 songs in disco genre...
Processing 100 songs in hiphop genre...
Processing 100 songs in jazz genre...
Processing 100 songs in metal genre...
Processing 100 songs in pop genre...
Processing 100 songs in reggae genre...
Processing 100 songs in rock genre...

```

Gambar 6.12 Fungsi Generate Features and Label Load Dataset Genre

6. Kenapa harus dilakukan pemisahan data training dan dataset sebesar 80%

Code Yang Digunakan :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:26:39 2020
4
5 @author: User

```

```

6 """
7
8 training_split = 0.8
9
10 alldata = np.column_stack((features, labels))
11
12 np.random.shuffle(alldata)
13 splitidx = int(len(alldata) * training_split)
14 train, test = alldata[:splitidx, :], alldata[splitidx:, :]
15
16 print(np.shape(train))
17 print(np.shape(test))
18
19 train_input = train[:, :-10]
20 train_labels = train[:, -10:]
21
22 test_input = test[:, :-10]
23 test_labels = test[:, -10:]
24
25 print(np.shape(train_input))
26 print(np.shape(train_labels))

```

▪ Penjelasan:

Untuk kemudahan dalam melakukan pengacakan, sebesar 80% untuk data training dan 20% untuk data test. Sehingga memperlihatkan bahwa data dipisah dan dipecah berpatokan dengan ketentuan 80%. Untuk hasil pertama data trainingnya ada 800 baris dengan 25000 kolom dan data set sebanyak 200 baris dengan 10 kolom, sedangkan untuk hasil kedua yang telah digabungkan dengan one-hot encoding maka data training terdapat 800 baris dan data set dengan 200 baris namun keduanya memiliki jumlah kolom yang sama yaitu 25010.

▪ Ilustrasi Gambar:

7. Parameter dari fungsi Sequensial()

Code Yang Digunakan :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:27:00 2020
4
5 @author: User
6 """
7
8 model = Sequential([
9     Dense(100, input_dim=np.shape(train_input)[1]),
10    Activation('relu'),
11    Dense(10),
12    Activation('softmax'),
13])

```

▪ Penjelasan:

```
In [7]: training_split = 0.8

In [8]: alldata = np.column_stack((features, labels))

In [9]: np.random.shuffle(alldata)
...: splitidx = int(len(alldata) * training_split)
...: train, test = alldata[:splitidx,:], alldata[splitidx:,:]

In [10]: print(np.shape(train))
...: print(np.shape(test))
(800, 25010)
(200, 25010)

In [11]: train_input = train[:, :-10]
...: train_labels = train[:, -10:]

In [12]: test_input = test[:, :-10]
...: test_labels = test[:, -10:]

In [13]: print(np.shape(train_input))
...: print(np.shape(train_labels))
(800, 25000)
(800, 10)
```

Gambar 6.13 Pemisahan Data Training dan Dataset

Untuk layer pertama densenya dari 100 neuron kemudian untuk inputan activationnya menggunakan fungsi relu. Dense 10 mengkategorikan 10 neuron untuk jenis genrenya untuk keluarannya menggunakan aktivasi yaitu fungsi Softmax.

- Ilustrasi Gambar:

```
In [14]: model = Sequential([
...:     Dense(100, input_dim=np.shape(train_input)[1]),
...:     Activation('relu'),
...:     Dense(10),
...:     Activation('softmax'),
...: ])
WARNING:tensorflow:From E:\Anaconda3\lib\site-packages\tensorflow\python\framework
\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and
will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
```

Gambar 6.14 Parameter Fungsi Sequensial

8. Parameter dari fungsi Compile().

Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:27:20 2020
4
5 @author: User
6 """
7
```

```

8 model.compile(optimizer='adam',
9                 loss='categorical_crossentropy',
10                metrics=['accuracy'])
11 print(model.summary())

```

- Penjelasan:

Untuk dilakukan pemrosesan menggunakan algortima adam sebagai optimizer yang sudah didefinisikan. Kemudian adam tersebut merupakan algoritma pengoptimalan dan untuk memperbarui bobot jaringan yang berulang berdasarkan data training sebelumnya. Untuk loss sendiri menggunakan categorical crossentropy yang difungsikan sebagai optimasi skor atau accuracy. Dan model tersebut digabungkan serta disimpulkan kemudian dicetak.

- Ilustrasi Gambar:

```

In [15]: model.compile(optimizer='adam',
...:                     loss='categorical_crossentropy',
...:                     metrics=['accuracy'])
...: print(model.summary())

Layer (type)                 Output Shape              Param #
=====
dense_1 (Dense)           (None, 100)             2500100
activation_1 (Activation)   (None, 100)               0
dense_2 (Dense)           (None, 10)                1010
activation_2 (Activation)   (None, 10)                0
=====
Total params: 2,501,110
Trainable params: 2,501,110
Non-trainable params: 0
=====
None

```

Gambar 6.15 Parameter Fungsi Compile

9. Parameter dari fungsi Fit().

Code Yang Digunakan :

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:27:36 2020
4
5 @author: User
6 """
7
8 model.fit(train_input, train_labels, epochs=10, batch_size=32,
9           validation_split=0.2)

```

- Penjelasan:

Untuk dilakukan pelatihan dengan epoch dengan rambatan balik sebanyak 10, kemudian dalam sekali epochs dilakukan 32 sampel yang diproses sebelum model diperbarui. Dilakukan validation split sebesar 20% untuk melakukan pengecekan pada cross score validation yang telah dilakukan.

- Ilustrasi Gambar:

```
In [47]: model.fit(train_input, train_labels, epochs=10, batch_size=32,
...:           validation_split=0.2)
Train on 640 samples, validate on 160 samples
Epoch 1/10
640/640 [=====] - 5s 8ms/step - loss: 2.1933 - acc: 0.2500 - val_loss:
1.8094 - val_acc: 0.3187
Epoch 2/10
640/640 [=====] - 2s 4ms/step - loss: 1.4572 - acc: 0.4844 - val_loss:
1.6058 - val_acc: 0.4125
Epoch 3/10
640/640 [=====] - 2s 3ms/step - loss: 1.0529 - acc: 0.6844 - val_loss:
1.5243 - val_acc: 0.4437
Epoch 4/10
640/640 [=====] - 2s 3ms/step - loss: 0.8013 - acc: 0.8125 - val_loss:
1.4311 - val_acc: 0.5375
Epoch 5/10
640/640 [=====] - 2s 3ms/step - loss: 0.6804 - acc: 0.8172 - val_loss:
1.5196 - val_acc: 0.4125
Epoch 6/10
640/640 [=====] - 2s 3ms/step - loss: 0.5424 - acc: 0.8797 - val_loss:
1.4561 - val_acc: 0.5437
Epoch 7/10
640/640 [=====] - 2s 3ms/step - loss: 0.4272 - acc: 0.9172 - val_loss:
1.5381 - val_acc: 0.4938
Epoch 8/10
640/640 [=====] - 2s 3ms/step - loss: 0.3408 - acc: 0.9531 - val_loss:
1.4135 - val_acc: 0.5437
Epoch 9/10
640/640 [=====] - 2s 3ms/step - loss: 0.2694 - acc: 0.9734 - val_loss:
1.5703 - val_acc: 0.4750
```

Gambar 6.16 Parameter Fungsi Fit

10. Parameter dari fungsi Evaluate()

Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:27:57 2020
4
5 @author: User
6 """
7
8 loss , acc = model.evaluate(test_input , test_labels , batch_size
9 =32)
10 print("Done!")
11 print("Loss: %.4f, accuracy: %.4f" % (loss , acc))
```

- Penjelasan:

Untuk menggunakan test input dan test label dilakukan evaluasi atau proses menemukan model terbaik yang mewakili data dan seberapa baik model yang dipilih akan dijalankan kedepannya. Kemudian pada hasilnya sendiri dapat dilihat bahwa Loss merupakan hasil prediksi yang salah sebanyak 1,7985 dan keakurasiannya prediksinya sebesar 0,4200.

- Ilustrasi Gambar:

```
In [48]: loss, acc = model.evaluate(test_input, test_labels, batch_size=32)
200/200 [=====] - 0s 1ms/step

In [49]: print("Done!")
...: print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
Done!
Loss: 1.3712, accuracy: 0.5300
```

Gambar 6.17 Parameter Fungsi Evaluate

11. Parameter dari fungsi Predict()

Code Yang Digunakan :

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr  6 23:28:16 2020
4
5 @author: User
6 """
7
8 model.predict(test_input[:1])
```

- Penjelasan:

Untuk melakukan prediksi diambil dari satu baris berdasarkan test_input . Nilai yang tertinggi terdapat pada label kedua yang dipilih prediksi yang tepat kemudian akan dikelompokkan.

- Ilustrasi Gambar:

```
In [50]: model.predict(test_input[:1])
Out[50]:
array([[1.14100585e-02, 5.04052186e-05, 9.51231807e-04, 2.89004028e-01,
       1.89829752e-01, 7.78380111e-02, 3.21492068e-02, 8.59113061e-06,
       3.85905296e-01, 1.28533337e-02]], dtype=float32)
```

Gambar 6.18 Parameter Fungsi Predict

12. PENANGANAN ERROR

- Ilustrasi Gambar:
- Penjelasan:

Install library tersebut pada Anaconda Prompt, jika proses penginstalan telah berhasil atau selesai dilakukan maka error akan teratasi.

```
...: import glob
...: import numpy as np
...: import matplotlib.pyplot as plt
...: from keras.models import Sequential
...: from keras.layers import Dense, Activation
...: from keras.utils.np_utils import to_categorical
Traceback (most recent call last):
File "<ipython-input-1-3bc11cc81b24>", line 1, in <module>
    import librosa
ModuleNotFoundError: No module named 'librosa'
```

Gambar 6.19 Error

BAB 7

CHAPTER 7

7.1 1174006 - Kadek Diva Krishna Murti

7.1.1 Teori

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

File teks harus dilakukan tokenizer untuk mengkonversi teks menjadi urutan integer indeks kata atau vektor binary, word count atau tf-idf.

Contoh:

```
texts = ["Diva makan nasi goreng."]
```

Hasil:

```
indexs: 'budi': 3, 'nasi': 1, 'makan': 2, 'goreng': 4
```

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1. dilengkapi dengan ilustrasi atau gambar.

```
1 kfolds = StratifiedKFold(n_splits=5)
2 splits = kfolds.split(d, d['CLASS'])
```

Konsep dasar dalam K Fold Cross Validation, yaitu dataset nantinya dibagi menjadi sejumlah K-buah partisi secara acak. Kemudian dilakukan sejumlah K-kali eksperimen, dimana masing-masing eksperimen menggunakan data partisi ke-K sebagai data testing dan memanfaatkan sisa partisi lainnya sebagai data training.

Pada Listing diatas cara kerjanya yaitu memanggil class StratifiedKFold() lalu tentukan K-nya sebanyak 5 (n_splits=5). Kemudian data KFold di split sesuai dengan atribut class.

3. Jelaskan apa maksudnya kode program for train, test in splits.dilengkapi dengan ilustrasi atau gambar.
Kode program for train, test in splits: maksudnya dilakukan perulangan dari data splits dengan variabel train dan test.
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.
Kode program train_content = d['CONTENT'].iloc[train_idx] maksudnya mengambil data pada atribut content dimana indexnya sesuai dengan train_idx.
Kode program test_content = d['CONTENT'].iloc[test_idx] maksudnya mengambil data pada atribut content dimana indexnya sesuai dengan test_idx.
5. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num_words=2000) dan tokenizer.fit_on_texts(train_content), dilengkapi dengan ilustrasi atau gambar.
Fungsi tokenizer = Tokenizer(num_words=2000) maksudnya memanggil class Tokenizer dengan parameter jumlah kata maksimum untuk disimpan, berdasarkan pada frekuensi kata yaitu 2000.
Fungsi tokenizer.fit_on_texts(train_content) maksudnya untuk metraining data text dengan parameter train_content.
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.
Fungsi d_train_inputs = tokenizer.texts_to_matrix(train_content, mode='tfidf') maksudnya mengkoversi list text ke matrix dengan parameter train_content dan mode tfidf.
Fungsi d_test_inputs = tokenizer.texts_to_matrix(test_content, mode='tfidf') maksudnya mengkoversi list text ke matrix dengan parameter test_content dan mode tfidf.
7. Jelaskan apa maksud dari fungsi d_train_inputs = d_train_inputs/np.absolute(d_train_inputs).max() dan d_test_inputs = d_test_inputs/np.absolute(d_test_inputs).max(), dilengkapi dengan ilustrasi atau gambar.
Fungsi d_train_inputs = d_train_inputs/np.absolute(d_train_inputs).max() maksudnya membagi hasil tfidf dengan nilai maxnya pada d_train_inputs.
Fungsi d_test_inputs = d_test_inputs/np.absolute(d_test_inputs).max() mak-

sudnya membagi hasil tfidf dengan nilai maxnya pada d_test_inputs

8. Jelaskan apa maksud fungsi dari `d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[0:train_idx])` dan `d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx:test_idx])` dalam kodeprogram, dilengkapi dengan ilustrasi atau gambar.
Fungsi `d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc[0:train_idx])` maksudnya mengubah vektor kelas (integer) ke matriks kelas biner sesuai dengan index classnya dari train_idx.
Fungsi `d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[train_idx:test_idx])` maksudnya mengubah vektor kelas (integer) ke matriks kelas biner sesuai dengan index classnya dari text_idx.

9. Jelaskan apa maksud dari fungsi di listing 7.2. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.

```

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

Maksud fungsi dari listing diatas adalah:

- Mendefinisikan class Sequential()
- Memanggil fungsi Dense dengan output array shape 512 dan input array shape 2000
- Memanggil fungsi Activation tipe relu
- Memanggil fungsi Dropout function input 0.5
- Memanggil fungsi Dense dengan output array shape 2
- Memanggil fungsi Activation tipe softmax

10. Jelaskan apa maksud dari fungsi di listing 7.3 dengan parameter tersebut.

```

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

Fungsi dari listing diatas maksudnya mengkonfigurasi model untuk dilakukan training.

11. Jelaskan apa itu Deep Learning. Deep learning merupakan metode pembelajaran yang dilakukan oleh mesin dengan cara meniru bagaimana sistem dasar otak manusia bekerja. Deep learning adalah salah satu cabang dari ilmu pembelajaran mesin (Machine Learning) yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam.

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning. Deep Neural Network adalah jaringan saraf tiruan yang memiliki beberapa lapisan antara lapisan input dan lapisan output. Perbedaan dari Deep Neural Network dan Deep Learning adalah Deep Neural Network merupakan metode yang digunakan sebagai pembelajaran oleh 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.

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.

```

1 # In[1]: import lib
2 import keras.models
3 import time
4 import keras.callbacks
5 from keras.layers import Conv2D, MaxPooling2D
6 from keras.layers import Dense, Dropout, Flatten
7 from keras.models import Sequential
8 from sklearn.preprocessing import OneHotEncoder
9 from sklearn.preprocessing import LabelEncoder
10 import numpy as np
11 import random
12 import csv
13 import tensorflow as tf
14 from PIL import Image as pil_image
15 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.

```

1 # In[2]: load all images (as numpy arrays) and save their classes
2
3 imgs = []
4 classes = []
5 with open('HASYv2/hasy-data-labels.csv') as csvfile:
6     csvreader = csv.reader(csvfile)
7     i = 0
8     for row in csvreader:
9         if i > 0:
10             img = keras.preprocessing.image.img_to_array(
11                 pil_image.open("HASYv2/" + row[0]))
12             # neuron activation functions behave best when input
13             # values are between 0.0 and 1.0 (or -1.0 and 1.0),
14             # so we rescale each pixel value to be in the range
15             # 0.0 to 1.0 instead of 0-255

```

```

14         img /= 255.0
15         imgs.append((row[0], row[2], img))
16         classes.append(row[2])
17         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.

```

1 # In [3]: shuffle the data, split into 80% train, 20% test
2
3 random.shuffle(imgs)
4 split_idx = int(0.8 * len(imgs))
5 train = imgs[:split_idx]
6 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.

```

1 # In [4]:
2
3
4 train_input = np.asarray(list(map(lambda row: row[2], train)))
5 test_input = np.asarray(list(map(lambda row: row[2], test)))
6
7 train_output = np.asarray(list(map(lambda row: row[1], train)))
8 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.

```

1 # In [5]: import encoder and one hot

```

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.

```

1 # In [6]: convert class names into one-hot encoding
2
3 # first, convert class names into integers
4 label_encoder = LabelEncoder()
5 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.

```

1 # In[7]: then convert integers into one-hot encoding
2 onehot_encoder = OneHotEncoder(sparse=False)
3 integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
4 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.

```

1 # In[8]: convert train and test output to one-hot
2 train_output_int = label_encoder.transform(train_output)
3 train_output = onehot_encoder.transform(
4     train_output_int.reshape(len(train_output_int), 1))
5 test_output_int = label_encoder.transform(test_output)
6 test_output = onehot_encoder.transform(
7     test_output_int.reshape(len(test_output_int), 1))
8
9 num_classes = len(label_encoder.classes_)
10 print("Number of classes: %d" % num_classes)

```

Number of classes: 369

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.

```

1 # In[9]: import sequential

```

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.

```

1 # In[10]: desain jaringan
2 model = tf.keras.Sequential()
3 model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3),
4     activation='relu',
5     input_shape=np.shape(train_input[0])))
6 model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
7 model.add(tf.keras.layers.Conv2D(32, (3, 3), activation='relu'))
8 model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
9 model.add(tf.keras.layers.Flatten())
10 model.add(tf.keras.layers.Dense(1024, activation='tanh'))
11 model.add(tf.keras.layers.Dropout(0.5))
12 model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))

```

```

12
13 model.compile(loss='categorical_crossentropy', optimizer='adam',
14                 metrics=['accuracy'])
15
16 print(model.summary())

```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_1 (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_2 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 32)	0
flatten_1 (Flatten)	(None, 1152)	0
dense_1 (Dense)	(None, 1024)	1180672
dropout_1 (Dropout)	(None, 1024)	0
dense_2 (Dense)	(None, 369)	378225
<hr/>		
Total params: 1,569,041		
Trainable params: 1,569,041		
Non-trainable params: 0		
<hr/>		
None		

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.

```

1 # In[11]: import sequential
2
3 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.

```

1 # In[12]: 5menit kali 10 epoch = 50 menit
2 model.fit(train_input, train_output,
3            batch_size=32,
4            epochs=10,
5            verbose=2,
6            validation_split=0.2,
7            callbacks=[tensorboard])
8
9 score = model.evaluate(test_input, test_output, verbose=2)
10 print('Test loss:', score[0])
11 print('Test accuracy:', score[1])

```

```

Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 54s - loss: 1.5568 - acc: 0.6243 - val_loss: 0.9898 - val_acc: 0.7257
Epoch 2/10
- 52s - loss: 0.9820 - acc: 0.7281 - val_loss: 0.8964 - val_acc: 0.7501
Epoch 3/10
- 52s - loss: 0.8730 - acc: 0.7523 - val_loss: 0.8776 - val_acc: 0.7531
Epoch 4/10
- 52s - loss: 0.8067 - acc: 0.7662 - val_loss: 0.8391 - val_acc: 0.7629
Epoch 5/10
- 52s - loss: 0.7520 - acc: 0.7771 - val_loss: 0.8406 - val_acc: 0.7579
Epoch 6/10
- 52s - loss: 0.7137 - acc: 0.7868 - val_loss: 0.8607 - val_acc: 0.7586
Epoch 7/10
- 52s - loss: 0.6812 - acc: 0.7922 - val_loss: 0.8696 - val_acc: 0.7648
Epoch 8/10
- 52s - loss: 0.6544 - acc: 0.7984 - val_loss: 0.8581 - val_acc: 0.7655
Epoch 9/10
- 52s - loss: 0.6312 - acc: 0.8015 - val_loss: 0.8518 - val_acc: 0.7595
Epoch 10/10
- 52s - loss: 0.6125 - acc: 0.8076 - val_loss: 0.8854 - val_acc: 0.7609
Test loss: 0.886258037221
Test accuracy: 0.762207626241

```

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.

```

1 # In[13]: try various model configurations and parameters to find
2      the best
3
4 results = []
5 for conv2d_count in [1, 2]:
6     for dense_size in [128, 256, 512, 1024, 2048]:
7         for dropout in [0.0, 0.25, 0.50, 0.75]:
8             model = tf.keras.Sequential()
9             for i in range(conv2d_count):
10                 if i == 0:
11                     model.add(tf.keras.layers.Conv2D(32,
12                         kernel_size=(3, 3), activation='relu', input_shape=np.
13                         shape(train_input[0])))
14                 else:
15                     model.add(tf.keras.layers.Conv2D(32,
16                         kernel_size=(3, 3), activation='relu'))
17                     model.add(tf.keras.layers.MaxPooling2D(pool_size
18                         =(2, 2)))
19                     model.add(tf.keras.layers.Flatten())
20                     model.add(tf.keras.layers.Dense(dense_size,
21                         activation='tanh'))
22                     if dropout > 0.0:
23                         model.add(tf.keras.layers.Dropout(dropout))
24                         model.add(tf.keras.layers.Dense(num_classes,
25                         activation='softmax'))
26
27             model.compile(loss='categorical_crossentropy',
28                           optimizer='adam', metrics=['accuracy'])
29
30 log_dir = '.\\logs\\conv2d_%d-dense_%d-dropout_%2f'
31 %
32     conv2d_count, dense_size, dropout)

```

```

27     tensorboard = keras.callbacks.TensorBoard(log_dir=
28         log_dir)
29
30         start = time.time()
31         model.fit(train_input, train_output, batch_size=32,
32         epochs=10,
33             verbose=0, validation_split=0.2, callbacks
34             =[tensorboard])
35             score = model.evaluate(test_input, test_output,
36             verbose=2)
37             end = time.time()
38             elapsed = end - start
39             print("Conv2D count: %d, Dense size: %d, Dropout: %.2f
40 f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" %
41             (conv2d_count, dense_size, dropout, score[0],
42             score[1], elapsed))
43             results.append((conv2d_count, dense_size, dropout,
44             score[0], score[1], elapsed))

```

```

Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.16, Accuracy: 0.74, Time: 419 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.92, Accuracy: 0.76, Time: 447 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.82, Accuracy: 0.77, Time: 452 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.79, Accuracy: 0.77, Time: 458 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.30, Accuracy: 0.74, Time: 430 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.12, Accuracy: 0.76, Time: 459 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.96, Accuracy: 0.77, Time: 461 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.75 - Loss: 0.78, Accuracy: 0.78, Time: 461 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.00 - Loss: 1.60, Accuracy: 0.74, Time: 440 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.25 - Loss: 1.43, Accuracy: 0.75, Time: 466 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.50 - Loss: 1.24, Accuracy: 0.75, Time: 471 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.75 - Loss: 0.87, Accuracy: 0.77, Time: 475 sec
Conv2D count: 1, Dense size: 1024, Dropout: 0.00 - Loss: 2.13, Accuracy: 0.72, Time: 480 sec
Conv2D count: 1, Dense size: 1024, Dropout: 0.25 - Loss: 1.94, Accuracy: 0.73, Time: 517 sec
Conv2D count: 1, Dense size: 1024, Dropout: 0.50 - Loss: 1.59, Accuracy: 0.73, Time: 526 sec
Conv2D count: 1, Dense size: 1024, Dropout: 0.75 - Loss: 0.98, Accuracy: 0.76, Time: 527 sec
Conv2D count: 1, Dense size: 2048, Dropout: 0.00 - Loss: 2.00, Accuracy: 0.70, Time: 587 sec
Conv2D count: 1, Dense size: 2048, Dropout: 0.25 - Loss: 2.02, Accuracy: 0.70, Time: 629 sec
Conv2D count: 1, Dense size: 2048, Dropout: 0.50 - Loss: 1.95, Accuracy: 0.72, Time: 639 sec
Conv2D count: 1, Dense size: 2048, Dropout: 0.75 - Loss: 1.29, Accuracy: 0.73, Time: 636 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.00 - Loss: 0.87, Accuracy: 0.76, Time: 531 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.25 - Loss: 0.79, Accuracy: 0.77, Time: 570 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.50 - Loss: 0.74, Accuracy: 0.78, Time: 568 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.75 - Loss: 0.79, Accuracy: 0.77, Time: 573 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.00 - Loss: 0.99, Accuracy: 0.76, Time: 550 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.25 - Loss: 0.85, Accuracy: 0.77, Time: 583 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.50 - Loss: 0.77, Accuracy: 0.78, Time: 579 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.75 - Loss: 0.76, Accuracy: 0.78, Time: 587 sec
Conv2D count: 2, Dense size: 512, Dropout: 0.00 - Loss: 1.18, Accuracy: 0.75, Time: 554 sec
Conv2D count: 2, Dense size: 512, Dropout: 0.25 - Loss: 1.07, Accuracy: 0.75, Time: 583 sec
Conv2D count: 2, Dense size: 512, Dropout: 0.50 - Loss: 0.84, Accuracy: 0.77, Time: 576 sec
Conv2D count: 2, Dense size: 512, Dropout: 0.75 - Loss: 0.77, Accuracy: 0.78, Time: 587 sec
Conv2D count: 2, Dense size: 1024, Dropout: 0.00 - Loss: 1.14, Accuracy: 0.74, Time: 557 sec
Conv2D count: 2, Dense size: 1024, Dropout: 0.25 - Loss: 0.94, Accuracy: 0.76, Time: 594 sec
Conv2D count: 2, Dense size: 1024, Dropout: 0.50 - Loss: 0.86, Accuracy: 0.76, Time: 599 sec
Conv2D count: 2, Dense size: 1024, Dropout: 0.75 - Loss: 0.81, Accuracy: 0.77, Time: 602 sec
Conv2D count: 2, Dense size: 2048, Dropout: 0.00 - Loss: 1.03, Accuracy: 0.75, Time: 572 sec
Conv2D count: 2, Dense size: 2048, Dropout: 0.25 - Loss: 1.09, Accuracy: 0.74, Time: 612 sec
Conv2D count: 2, Dense size: 2048, Dropout: 0.50 - Loss: 0.97, Accuracy: 0.74, Time: 616 sec
Conv2D count: 2, Dense size: 2048, Dropout: 0.75 - Loss: 0.85, Accuracy: 0.76, Time: 619 sec

```

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.

```

1 # In[14]: rebuild/retrain a model with the best parameters (from
2     the search) and use all data
3 model = tf.keras.Sequential()
4 model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3),
5     activation='relu',
6         input_shape=np.shape(train_input[0])))
7 model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
8 model.add(tf.keras.layers.Conv2D(32, (3, 3), activation='relu'))

```

```

7 model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
8 model.add(tf.keras.layers.Flatten())
9 model.add(tf.keras.layers.Dense(128, activation='tanh'))
10 model.add(tf.keras.layers.Dropout(0.5))
11 model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
12 model.compile(loss='categorical_crossentropy',
13                 optimizer='adam', metrics=['accuracy'])
14 print(model.summary())

```

Layer (type)	Output Shape	Param #
conv2d_68 (Conv2D)	(None, 30, 30, 32)	896
max_pooling2d_68 (MaxPooling)	(None, 15, 15, 32)	0
conv2d_69 (Conv2D)	(None, 13, 13, 32)	9248
max_pooling2d_69 (MaxPooling)	(None, 6, 6, 32)	0
flatten_45 (Flatten)	(None, 1152)	0
dense_89 (Dense)	(None, 128)	147584
dropout_34 (Dropout)	(None, 128)	0
dense_90 (Dense)	(None, 369)	47601

Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0

None
Epoch 1/10
- 80s - loss: 1.8212 - acc: 0.5797
Epoch 2/10
- 77s - loss: 1.0929 - acc: 0.7030
Epoch 3/10
- 77s - loss: 0.9790 - acc: 0.7278
Epoch 4/10
- 77s - loss: 0.9189 - acc: 0.7408
Epoch 5/10
- 77s - loss: 0.8839 - acc: 0.7479
Epoch 6/10
- 77s - loss: 0.8560 - acc: 0.7528
Epoch 7/10
- 77s - loss: 0.8360 - acc: 0.7582
Epoch 8/10
- 77s - loss: 0.8161 - acc: 0.7610
Epoch 9/10
- 77s - loss: 0.8020 - acc: 0.7653
Epoch 10/10
- 77s - loss: 0.7913 - acc: 0.7657

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.

```

1 # In[15]: join train and test data so we train the network on all
2 # data we have available to us
3 model.fit(np.concatenate((train_input, test_input)),
4           np.concatenate((train_output, test_output)),
5           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.

```
1 # In[16]: save the trained model
2 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.

```
1 # In[17]: save label encoder (to reverse one-hot encoding)
2 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.

```
1 # In[18]: load the pre-trained model and predict the math symbol
          for an arbitrary image;
2 # the code below could be placed in a separate file
3
4 model2 = keras.models.load_model("mathsymbols.model")
5 print(model2.summary())
```

Layer (type)	Output Shape	Param #
<hr/>		
conv2d_68 (Conv2D)	(None, 30, 30, 32)	896
<hr/>		
max_pooling2d_68 (MaxPooling)	(None, 15, 15, 32)	0
<hr/>		
conv2d_69 (Conv2D)	(None, 13, 13, 32)	9248
<hr/>		
max_pooling2d_69 (MaxPooling)	(None, 6, 6, 32)	0
<hr/>		
flatten_45 (Flatten)	(None, 1152)	0
<hr/>		
dense_89 (Dense)	(None, 128)	147584
<hr/>		
dropout_34 (Dropout)	(None, 128)	0
<hr/>		
dense_90 (Dense)	(None, 369)	47601
<hr/>		
Total params:	205,329	
Trainable params:	205,329	
Non-trainable params:	0	
<hr/>		
	None	

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.

```
1 # In[19]: restore the class name to integer encoder
2 label_encoder2 = LabelEncoder()
3 label_encoder2.classes_ = np.load('classes.npy')
4
5
6 def predict(img_path):
7     newimg = keras.preprocessing.image.img_to_array(pil_image.
8         open(img_path))
9     newimg /= 255.0
10
11     # do the prediction
12     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
13
14     # figure out which output neuron had the highest score, and
15     # reverse the one-hot encoding
16     inverted = label_encoder2.inverse_transform(
17         [np.argmax(prediction)]) # argmax finds highest-scoring
18     output
19     print("Prediction: %s, confidence: %.2f" %
20           (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.

```
1 # In [20]: grab an image (we'll just use a random training image  
2     for demonstration purposes)  
3 predict("HASYv2/hasy-data/v2-00010.png")  
4 predict("HASYv2/hasy-data/v2-00500.png")  
5 predict("HASYv2/hasy-data/v2-00700.png")
```

```
Prediction: A, confidence: 0.87
Prediction: \pi, confidence: 0.58
Prediction: \alpha, confidence: 0.88
```

7.1.3 Penanganan Error

7.1.4 Bukti Tidak Plagiat



Gambar 7.1 Kecerdasan Buatan.

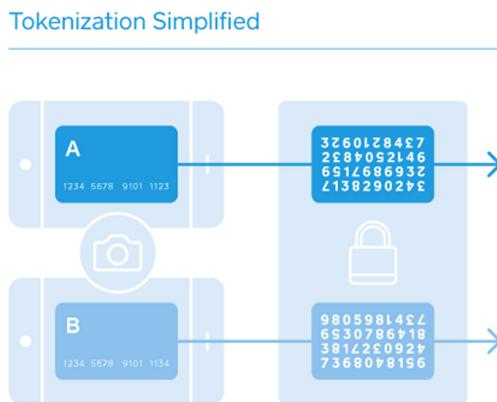
7.2 Muhammad Tomy Nur Maulidy 1174031

7.2.1 Teori

7.2.1.1 Soal No. 1 Kenapa file teks harus dilakukan tokenizer, dilengkapi dengan ilustrasi atau gambar.

Karena tokenizer ini berfungsi untuk mengkonversi teks menjadi urutan integer indeks kata atau vektor binary, word count atau tf-idf. Text harus dilakukan tokenizer agar dapat dirubah menjadi vektor. Dari perubahan ke vektor tersebut maka data/textnya dapat dibaca oleh komputer (terkomputerisasi).

- Ilustrasi Gambar:



Gambar 7.2 Tokenizer

7.2.1.2 Soal No. 2 Konsep dasar K Fold Cross Validation pada dataset komentar Youtube, dilengkapi dengan ilustrasi atau gambar.

Pada Starti

edKFold memiliki input untuk setiap class yang terbagi menjadi 5 class pada setiap class-nya. Lalu dimasukkan kedalam class dataset youtube.

- Ilustrasi Gambar:

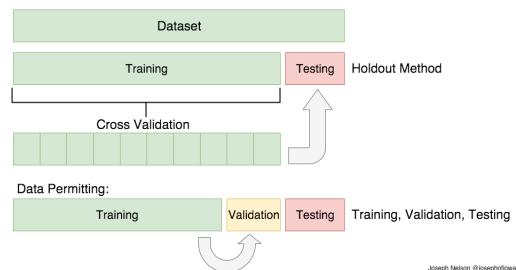
7.2.1.3 Soal No. 3 Maksud kode program for train dan test in splits, dilengkapi dengan ilustrasi atau gambar.

Untuk melakukan pengujian atas data pada dataset sudah di tidak terjadi penumpukan dan split. Dimana di setiap class tidak akan muncul id yang sama.

- Ilustrasi Gambar :

Gambar 1 – Skema 10 fold CV

Gambar 7.3 Konsep dasar K Fold Cross Validation

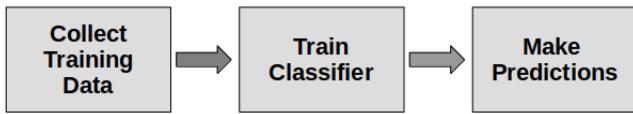


Gambar 7.4 Train dan Test in Split

7.2.1.4 Soal No. 4 Maksud kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx], dilengkapi dengan ilustrasi atau gambar.

Untuk mengambil data pada kolom CONTENT yang merupakan bagian dari train idx dan test idx.

- Ilustrasi Gambar:



Gambar 7.5 Train content

7.2.1.5 Soal No. 5 Maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar.

Yang pertama, yaitu fungsi tokenizer ialah mem-vektorisasi jumlah kata yang ingin diubah kedalam bentuk token 2000 kata.

Yang kedua, untuk melakukan fit tokenizer untuk dat trainnya dengan data test nya untuk kolom CONTENT saja.

- Ilustrasi Gambar :

Tokenization

is a process of breaking up a piece of **text** into many pieces, such as sentences and words. It works by separating **words** using spaces and punctuation.

```

[1]: from nltk.tokenize import sent_tokenize
      sentence = "I love ice cream. I also like steak."
      sent_tokenize(sentence)
  
```

```

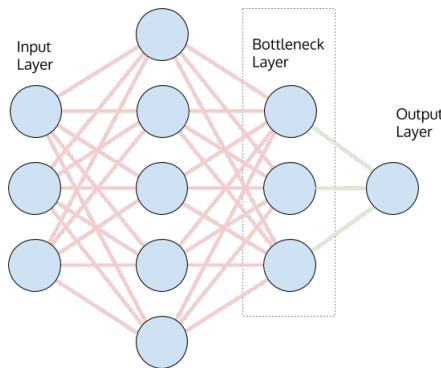
:[1]: ['I love ice cream.', 'I also like steak.']
  
```

Gambar 7.6 Tokenizer

7.2.1.6 Soal No. 6 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 atau gambar.

Variabel d train input untukmelakukan tokenizer dari bentuk teks ke matrix dari data train content dengan mode tfidf dan variabel d test inputs sama saja untuk data test.

- Ilustrasi Gambar:

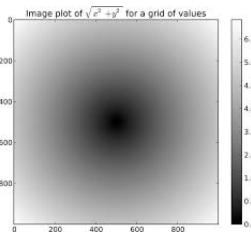


Gambar 7.7 Train Inputs 1

7.2.1.7 Soal No. 7 Maksud dari fungsi $d\text{ train inputs} = d\text{ train inputs}/\text{np.amax}(\text{np.absolute}(d\text{ train inputs}))$ dan $d\text{ test inputs} = d\text{ test inputs}/\text{np.amax}(\text{np.absolute}(d\text{ test inputs}))$, dilengkapi dengan ilustrasi atau gambar.

Akan membagi matrix tfidf dengan amax untuk mengembalikan array atau maksimum array. Kemudian hasilnya dimasukan dalam variabel $d\text{ train inputs}$ untuk data train dan $d\text{ test inputs}$ untuk data test dengan nominal bilangan tanpa ada bilangan negatif dan koma.

- Ilustrasi Gambar :

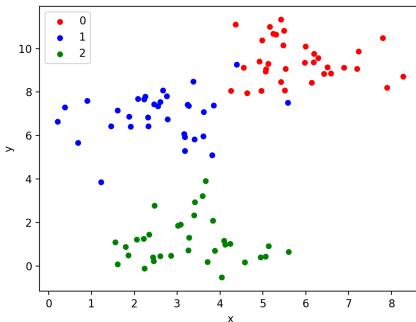


Gambar 7.8 Train Inputs 2

7.2.1.8 Soal No. 8 Maksud dari $d\text{ train outputs} = \text{np utils.to categorical}(d[\text{CLASS}].iloc[train idx])$ dan $d\text{ test outputs} = \text{np utils.to categorical}(d[\text{CLASS}].iloc[test idx])$

Fungsi pada kode program tersebut ditujukan untuk melakukan one-hot encoding supaya bisa masuk dan digunakan pada neural network. One-hot encoding diambil dari class yang berarti hanya terdapat 2 neuron, yaitu satu nol(1,0) atau nol satu(0,1) karena pilihannya hanya ada dua.

- Ilustrasi Gambar:



Gambar 7.9 Compile model

7.2.1.9 **Soal No. 9** Maksud dari listing 7.2.

Fungsi kode program tersebut untuk melakukan pemodelan dengan sequential, membandingkan setiap satu larik elemen dengan cara satu persatu secara beruntun. Terdapat 512 neuron inputan dengan input shape 2000 vektor yang sudah di-normalisasi. Lalu model dilakukan aktivasi dengan fungsi 'relu'. Kemudian pemotongan bobot supaya tidak overfitting sebesar 50 persen dari neuron inputan 512. Lalu pada layer output terdapat 2 neuron outputan yaitu nol (1,0) atau nol satu (0,1). Kemudian outputan tersebut diaktifasi menggunakan fungsi softmax.

7.2.1.10 **Soal No. 10** Maksud dari listing 7.3.

Fungsi kode program tersebut untuk model yang telah dibuat selanjutnya dicompile dengan menggunakan algoritma optimisasi, fungsi loss, dan fungsi metrik.

7.2.1.11 **Soal No. 11** Deep Learning

Deep learning, yang bisa diartikan sebagai rangkaian metode untuk melatih jaringan saraf buatan multi-lapisan. Ternyata, metode ini efektif dalam mengidentifikasi pola dari data. Manakala media membicarakan jaringan saraf, kemungkinan yang dimaksud adalah deep learning.

7.2.1.12 Soal No. 12 Deep Neural Network, dan apa bedanya dengan Deep Learning

Algoritma DNN (Deep Neural Networks) adalah salah satu algoritma berbasis jaringan saraf yang dapat digunakan untuk pengambilan keputusan. Contoh yang dibahas kali ini adalah mengenai penentuan penerimaan pengajuan kredit sepeda motor baru berdasarkan kelompok data yang sudah ada.

Pembedannya dengan Deep Learning adalah terletak pada kedalaman model, deep learning adalah frasa yang digunakan untuk jaringan saraf yang kompleks.

7.2.1.13 Soal No. 13 Jelaskan dengan ilustrasi gambar buatan sendiri, bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod3+1)x(NPM mod3+1) yang terdapat max pooling.(nilai 30)

Karena NPM saya 1164067 dan hasil dari $(NPM \bmod 3) + 1 = 2$, maka saya menggunakan matrik kernel berukuran 2×2 . Misalkan $f(x,y)$ yang digunakan berukuran 3×3 dan kernel atau mask berukuran 2×2 adalah sebagai berikut:

Gambar Matriks:

$$F(x,y) = \begin{matrix} 2 & 3 & 5 \\ 1 & 0 & 8 \\ 7 & 2 & 0 \end{matrix} \quad \begin{matrix} 1 & 0 \\ 2 & 4 \end{matrix}$$

Gambar 7.10 Perhitungan algoritma konvolusi

Penyelesaian dari operasi konvolusi antara $f(x,y)$ dengan kernel $g(x,y)$ adalah $f(x,y) * g(x,y)$

- Tempatkan matrik kernel di sebelah kiri atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(2 \times 1) + (3 \times 0) + (1 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

- Tempatkan matrik kernel di sebelah kanan atas, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(3 \times 1) + (5 \times 0) + (0 \times 2) + (8 \times 4)$$

Sehingga didapat hasil konvolusi = 35

- Tempatkan matrik kernel di sebelah kiri bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(1 \times 1) + (0 \times 0) + (7 \times 2) + (2 \times 4)$$

Sehingga didapat hasil konvolusi = 23

- Tempatkan matrik kernel di sebelah kanan bawah, lalu hitung nilai piksel pada posisi (0,0) dari kernel tersebut. Konvolusi dihitung dengan cara berikut :

$$(0 \times 1) + (8 \times 0) + (2 \times 2) + (0 \times 4)$$

Sehingga didapat hasil konvolusi = 4

Hasil:

$$\begin{matrix} & 4 & 35 \\ 23 & & 4 \end{matrix}$$

Gambar 7.11 Hasil

7.2.2 Praktek

- **Soal No. 1** Jelaskan arti dari setiap baris kode program pada blok # In[1] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:46:02 2020
4
5 @author: User
6 """
7
8 import csv
9 from PIL import Image as pil_image
10 import keras.preprocessing.image

```

- Penjelasan:

Baris Code 1: Memasukkan atau mengimport file csv

Baris Code 2: Memasukkan module image sebagai pil_image dari library PIL

Baris Code 3: Memasukkan atau mengimport fungsi keras.preprocessing.image

- Hasil output:

```

In [1]: import csv
...: from PIL import Image as pil_image
...: import keras.preprocessing.image
Using TensorFlow backend.

```

Gambar 7.12 1

7.2.2.2 Soal No. 2 Jelaskan arti dari setiap baris kode program pada blok # In[2] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:25 2020
4
5 @author: User
6 """
7
8 imgs = []
9 classes = []
10 with open('HASYv2/hasy-data-labels.csv') as csvfile:
11     csvreader = csv.reader(csvfile)
12     i = 0
13     for row in csvreader:
14         if i > 0:
15             img = keras.preprocessing.image.img_to_array(
16                 pil_image.open("HASYv2/" + row[0]))
17                 # neuron activation functions behave best when input
18                 # values are between 0.0 and 1.0 (or -1.0 and 1.0),
19                 # so we rescale each pixel value to be in the range
20                 # 0.0 to 1.0 instead of 0-255
21                 img /= 255.0
22             imgs.append((row[0], row[2], img))
23             classes.append(row[2])
24         i += 1

```

- Penjelasan:

Baris Code 1: Membuat variabel imgs tanpa parameter

Baris Code 2: Membuat variabel classes tanpa parameter

Baris Code 3: Membuka file HASYv2/hasy-data-labels.csv

Baris Code 4: Membuat variabel csvreader untuk pembacaan dari file csv yang dimasukkan

Baris Code 5: Membuat variabel i dengan parameter 0

Baris Code 6: Mengeksekusi baris dari pembacaan csv

Baris Code 7: Mengaplikasikan perintah "if" dengan ketentuan variabel i lebih besar dari angka 0, maka akan dilanjutkan ke perintah berikutnya

Baris Code 8: Membuat variabel img yang mengubah image menjadi bentuk array dari file HASYv2 yang dibuka dengan row berparameter 0.

Baris Code 9: Membuat variabel img atau dengan nilai 255.0

Baris Code 10: Mendefinisikan fungsi imgs.append dimana merupakan proses melampirkan atau menggabungkan data dengan file lain atau set data yang ditentukan dengan 3 parameter yaitu row[0], row[2] dan variabel img.

Baris Code 11: Mendefinisikan fungsi append kembali dari variabel classes dengan parameternya row[2].

Baris Code 12: Mendefinisikan fungsi dimana i variabel i akan ditambah nilainya sehingga akan bernilai 1.

- Hasil output:

```
In [2]: imgs = []
...: classes = []
...: with open('HASYv2/hasy-data-labels.csv') as csvfile:
...:     csvreader = csv.reader(csvfile)
...:     i = 0
...:     for row in csvreader:
...:         if i > 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
...:         img /= 255.0
...:         imgs.append((row[0], row[2], img))
...:         classes.append(row[2])
...:         i += 1
```

Gambar 7.13 2

7.2.2.3 Soal No. 3 Jelaskan arti dari setiap baris kode program pada blok # In[3] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import random
9 random.shuffle(imgs)
10 split_idx = int(0.8*len(imgs))
11 train = imgs[:split_idx]
12 test = imgs[split_idx:]
```

- Penjelasan:

Baris Code 1: Memasukkan module random

Baris Code 2: Melakukan pengocokan atau pengacakkan pada module random dengan parameter variabelnya imgs

Baris Code 3: Membagi dan memecah index dalam bentuk integer dengan mengkalikan nilai 0,8 dan fungsi len yang akan mengembalikan jumlah item dalam datanya dari variabel imgs

Baris Code 4: Membuat variabel train yang mengeksekusi imgs dengan pemecahan index awal pada data

Baris Code 5: Membuat variabel test yang mengeksekusi imgs dengan pemecahan index akhir pada data

- Hasil output:

```
In [3]: import random
...: random.shuffle(imgs)
...: split_idx = int(0.8*len(imgs))
...: train = imgs[:split_idx]
...: test = imgs[split_idx:]
```

Gambar 7.14 3

7.2.2.4 Soal No. 4 Jelaskan arti dari setiap baris kode program pada blok # In[4] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 import numpy as np
9
10 train_input = np.asarray(list(map(lambda row: row[2], train)))
11 test_input = np.asarray(list(map(lambda row: row[2], test)))
12
13 train_output = np.asarray(list(map(lambda row: row[1], train)))
14 test_output = np.asarray(list(map(lambda row: row[1], test)))
```

- Penjelasan:

Baris Code 1: Mengimport library numpy sebagai np

Baris Code 2: Membuat variabel train_input untuk input menjadi sebuah array dari np menggunakan fungsi list untuk mengkoleksikan data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungksikan lamda pada row dengan parameter [2] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 3: Membuat variabel test_input dengan fungsi seperti train_input yang membedakan hanya datanya atau inputan yang diproses berasal dari variabel test

Baris Code 4: Membuat variabel train_output untuk mengubah keluaran menjadi sebuah array dari np dengan menggunakan fungsi list untuk mengkoleksi

data yang dipilih dan diubah. Didalamnya diterapkan fungsi map untuk mengembalikan iterator dari datanya dengan memfungskan lamda pada row dengan parameter[1] untuk membuat objek fungsi menjadi lebih kecil dan mudah dieksekusi dari variabel train.

Baris Code 5: Membuat variabel test_output dengan fungsi yang sama seperti train_output yang membedakan hanya datanya atau inputan yang diproses berdasarkan dari variabel test

- Hasil output:

```
In [4]: 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 7.15 4

7.2.2.5 Soal No. 5 Jelaskan arti dari setiap baris kode program pada blok # In[5] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 from sklearn.preprocessing import LabelEncoder
9 from sklearn.preprocessing import OneHotEncoder
```

- Penjelasan:

Baris Code 1: Memasukkan modul atau fungsi LabelEncoder dari sklearn.preprocessing untuk dapat digunakan menormalkan label dimana label encoder didefinisikan dengan nilai antara 0 dan n_classes-1.

Baris Code 2: Memasukkan modul atau fungsi OneHotEncoder dari sklearn.preprocessing untuk mendefinisikan fitur input dimana mengambil nilai dalam kisaran [0, maks (nilai)).

- Hasil output:

```
In [5]: from sklearn.preprocessing import LabelEncoder
...: from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.16 5

7.2.2.6 Soal No. 6 Jelaskan arti dari setiap baris kode program pada blok # In[6] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 label_encoder = LabelEncoder()
9 integer_encoded = label_encoder.fit_transform(classes)

```

- Penjelasan:

Baris Code 1: Membuat variabel label_encoder dengan modul atau fungsi dari LabelEncoder tanpa parameter

Baris Code 2: Membuat variabel integer_encoded dengan fungsi label_encoder.fit_transform dari variabel classes yang akan mengembalikan beberapa data yang diubah kembali dari variabel label_encoder.

- Hasil output:

```
In [6]: label_encoder = LabelEncoder()
...: integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.17 6

7.2.2.7 Soal No. 7 Jelaskan arti dari setiap baris kode program pada blok # In[7] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:27 2020
4
5 @author: User
6 """
7
8 onehot_encoder = OneHotEncoder(sparse=False)
9 integer_encoded = integer_encoded.reshape(len(integer_encoded),
10 1)
onehot_encoder.fit(integer_encoded)

```

- Penjelasan:

Baris 1: Membuat variabel onehot_encoder yang memanggil fungsi OneHotEncoder tanpa mengembalikan matriks karena sparse=false.

Baris 2: Membuat variabel integer_encoded memanggil variabel integer_encoded pada kode program 6 untuk dieksekusi memberikan bentuk baru ke array tanpa mengubah datanya dari mengembalikan panjang nilai dari integer_encoded.

Baris 3: Onehotencoding melakukan fitting pada integer_encoded.

- Hasil output:

```
In [7]: onehot_encoder = OneHotEncoder(sparse=False)
...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
...: onehot_encoder.fit(integer_encoded)
D:\Anaconda\lib\site-packages\sklearn\preprocessing\_encoders.py:371: FutureWarning:
The handling of integer data will change in version 0.22. Currently, the categories
are determined based on the range [0, max(values)], while in the future they will be
determined based on the unique values.
If you want the future behaviour and silence this warning, you can specify
"categories='auto'".
In case you used a LabelEncoder before this OneHotEncoder to convert the categories
to integers, then you can now use the OneHotEncoder directly.
    warnings.warn(msg, FutureWarning)
Out[7]:
OneHotEncoder(categorical_features=None, categories=None,
               dtype=<class 'numpy.float64'>, handle_unknown='error',
               n_values=None, sparse=False)
```

Gambar 7.18 7

7.2.2.8 Soal No. 8 Jelaskan arti dari setiap baris kode program pada blok # In[8] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 train_output_int = label_encoder.transform(train_output)
9 train_output = onehot_encoder.transform(train_output_int.reshape(
10     len(train_output_int), 1))
11 test_output_int = label_encoder.transform(test_output)
12 test_output = onehot_encoder.transform(test_output_int.reshape(
13     len(test_output_int), 1))
14 num_classes = len(label_encoder.classes_)
15 print("Number of classes: %d" % num_classes)
```

- Penjelasan:

Baris 1: Membuat variabel train_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel train_output.

Baris 2: Membuat variabel train_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter train_output_int

yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari train_output_int telah dikembalikan.

Baris 3: Membuat variabel test_output_int yang mengeksekusi label_encoder dengan mengubah nilai dari parameter variabel test_output.

Baris 4: Membuat variabel test_output yang mengeksekusi variabel onehot_encoder dari kode program 7 dengan mengubah nilai dari variabel parameter test_output_int yang datanya sudah diubah kedalam bentuk array dan panjang nilai dari test_output_int telah dikembalikan.

Baris 5: Membuat variabel num_classes untuk mengetahui jumlah class dari label_encoder

Baris 6: Perintah print digunakan untuk memunculkan hasil dari variabel num_classes

```
In [8]: 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 7.19 8

- Hasil output:

7.2.2.9 Soal No. 9 Jelaskan arti dari setiap baris kode program pada blok # In[9] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:47:28 2020
4
5 @author: User
6 """
7
8 from keras.models import Sequential
9 from keras.layers import Dense, Dropout, Flatten
10 from keras.layers import Conv2D, MaxPooling2D
```

- Penjelasan:

Baris 1: Melakukan importing fungsi model sequential dari library keras.

Baris 2: Melakukan importing fungsi layer dense, dropout, dan flatten dari library keras.

Baris 3: Melakukan importing fungsi layer Conv2D dan MaxPooling2D dari library keras.

- Hasil output:

```
In [9]: from keras.models import Sequential
...: from keras.layers import Dense, Dropout, Flatten
...: from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.20 9

7.2.2.10 Soal No. 10 Jelaskan arti dari setiap baris kode program pada blok # In[10] dan hasil luarannya.

- Code:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                 input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(1024, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.add(Dense(num_classes, activation='softmax'))
18
19 model.compile(loss='categorical_crossentropy', optimizer='adam',
20                 metrics=['accuracy'])
21
22 print(model.summary())
```

- Penjelasan:

Baris 1: Melakukan pemodelan Sequential.

Baris 2: Menambahkan Konvolusi 2D dengan 32 filter konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu dengan data dari train input mulai dari baris nol.

Baris 3: Menambahkan Max Pooling dengan matriks 2x2.

Baris 4: Penambahan Konvolusi 2D dengan 32 filter, konvolusi masing-masing berukuran 3x3 dengan algoritam activation relu.

Baris 5 Menambahkan Max Pooling dengan matriks 2x2.

Baris 6: Mendefinisikan inputan dengan 1024 neuron dan menggunakan algoritma tanh untuk activationnya.

Baris 7: Dropout terdiri dari pengaturan secara acak tingkat pecahan unit input ke 0 pada setiap pembaruan selama waktu pelatihan, yang membantu mencegah overfitting sebesar 50% .

Baris 8: Untuk output layer menggunakan data dari variabel num classes dengan fungsi activationnya softmax.

Baris 9: Konfigurasi proses pembelajaran, melalui metode compile, sebelum melatih suatu model.

Baris 10: Menampilkan model yang telah dibuat.

- Hasil output:

```
...: model.add(Flatten())
...: model.add(Dense(1024, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
...: metrics=['accuracy'])
...
...: print(model.summary())
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\framework
\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is
deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\keras\backend
\tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with
keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 -
keep_prob`.
Layer (type)          Output Shape         Param #
=====
conv2d_1 (Conv2D)     (None, 30, 30, 32)      896
max_pooling2d_1 (MaxPooling2D) (None, 15, 15, 32)    0
conv2d_2 (Conv2D)     (None, 13, 13, 32)      9248
max_pooling2d_2 (MaxPooling2D) (None, 6, 6, 32)      0
flatten_1 (Flatten)   (None, 1152)           0
dense_1 (Dense)       (None, 1024)          1180672
dropout_1 (Dropout)   (None, 1024)           0
dense_2 (Dense)       (None, 369)            378225
=====
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
=====
None
```

Gambar 7.21 10

7.2.2.11 Soal No. 11 Jelaskan arti dari setiap baris kode pada blok # In[11] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8
9 import keras.callbacks
10 tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
    style')

```

- Penjelasan:

Baris Code 1: Melakukan import library keras.callbacks yang digunakan pada penulisan log untuk TensorBoard, untuk memvisualisasikan grafik dinamis dari pelatihan dan metrik pengujian.

Baris Code 2: Membuat variabel tensorboard untuk mendefinisikan fungsi TensorBoard pada keras.callbacks yang digunakan sebagai alat visualisasi yang disediakan dengan TensorFlow. Dan untuk fungsi log_dir memanggil data yaitu './logs/mnist-style'

- Hasil output:

```
In [11]: import keras.callbacks
...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
```

Gambar 7.22 11

7.2.2.12 Soal No. 12 Jelaskan arti dari setiap baris kode program pada blok # In[12] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:11 2020
4
5 @author: User
6 """
7
8 model.fit(train_input, train_output,
9            batch_size=32,
10           epochs=10,
11           verbose=2,
12           validation_split=0.2,
13           callbacks=[tensorboard])
14
15 score = model.evaluate(test_input, test_output, verbose=2)
16 print('Test loss:', score[0])
17 print('Test accuracy:', score[1])

```

- Penjelasan:

Baris Code 1: Menerapkan fungsi model.fit yang didalamnya memproses train_input, train_output

Baris Code 2: Penerapan fungsi yang sama difungsikan batch_size apabila batch_sizenya tidak ditemukan maka otomatis akan dijadikan nilai 32

Baris Code 3: Penerapan fungsi yang sama, difungsikan epochs dimana perulangan dari berapa kali nilai yang digunakan untuk data, dan jumlahnya ialah 10

Baris Code 4: Mendefinisikan fungsi verbose untuk digunakan sebagai opsi menghasilkan informasi logging dari data yang ditentukan dengan nilai 2

Baris Code 5: Mendefinisikan fungsi validation_split untuk memecah nilai dari perhitungan validasinya sebesar 0,2. (Fraksi data pelatihan untuk digunakan sebagai data validasi)

Baris Code 6: Mendefinisikan fungsi callbacks dengan parameternya yang mengeksekusi tensorboard dimana digunakan untuk visualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses

Baris Code 7: Mendefinisikan variabel score dengan fungsi evaluate dari model dengan parameter test_input, tst_output dan verbose=2 untuk memprediksi output dan input yang diberikan dan kemudian menghitung fungsi metrik yang ditentukan dalam modelnya

Baris Code 8: Mencetak score optimasi dari test dengan ketentuan nilai parameter 0

Baris Code 9: Mencetak score akurasi dari test dengan ketentuan nilai parameter 1

- Hasil output:

7.2.2.13 Soal No. 13 Jelaskan arti dari setiap baris kode program pada blok # In[13] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 import time
9
10 results = []
11 for conv2d_count in [1, 2]:
12     for dense_size in [128, 256, 512, 1024, 2048]:
13         for dropout in [0.0, 0.25, 0.50, 0.75]:
14             model = Sequential()

```

```
In [12]: model.fit(train_input, train_output,
...:     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])
WARNING:tensorflow:From D:\Anaconda\lib\site-packages\tensorflow\python\ops\numpy_ops.py:3066: to_int32 (from tensorflow.python.ops.numpy_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
- 2007s - loss: 1.5334 - acc: 0.6294 - val_loss: 0.9824 - val_acc: 0.7285
Epoch 2/10
- 918s - loss: 0.9694 - acc: 0.7321 - val_loss: 0.9162 - val_acc: 0.7494
Epoch 3/10
- 562s - loss: 0.8543 - acc: 0.7556 - val_loss: 0.8883 - val_acc: 0.7530
Epoch 4/10
- 503s - loss: 0.7854 - acc: 0.7699 - val_loss: 0.8441 - val_acc: 0.7666
Epoch 5/10
- 361s - loss: 0.7364 - acc: 0.7796 - val_loss: 0.8491 - val_acc: 0.7663
Epoch 6/10
- 366s - loss: 0.6936 - acc: 0.7899 - val_loss: 0.8522 - val_acc: 0.7692
Epoch 7/10
- 360s - loss: 0.6591 - acc: 0.7962 - val_loss: 0.8580 - val_acc: 0.7668
Epoch 8/10
- 389s - loss: 0.6344 - acc: 0.8017 - val_loss: 0.8496 - val_acc: 0.7654
Epoch 9/10
- 357s - loss: 0.6076 - acc: 0.8072 - val_loss: 0.8597 - val_acc: 0.7638
Epoch 10/10
- 359s - loss: 0.5905 - acc: 0.8116 - val_loss: 0.8889 - val_acc: 0.7637
Test loss: 0.8794204677150739
Test accuracy: 0.7632181175230488
```

Gambar 7.23 12

```

15         for i in range(conv2d_count):
16             if i == 0:
17                 model.add(Conv2D(32, kernel_size=(3, 3),
18 activation='relu', input_shape=np.shape(train_input[0])))
19             else:
20                 model.add(Conv2D(32, kernel_size=(3, 3),
21 activation='relu'))
22                 model.add(MaxPooling2D(pool_size=(2, 2)))
23                 model.add(Flatten())
24                 model.add(Dense(dense_size, activation='tanh'))
25                 if dropout > 0.0:
26                     model.add(Dropout(dropout))
27                 model.add(Dense(num_classes, activation='softmax'))
28
29                 model.compile(loss='categorical_crossentropy',
30 optimizer='adam', metrics=['accuracy'])
31
32                 log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
33 (conv2d_count, dense_size, dropout)
34                 tensorboard = keras.callbacks.TensorBoard(log_dir=
log_dir)
35
36                 start = time.time()
37                 model.fit(train_input, train_output, batch_size=32,
38 epochs=10,
39 verbose=0, validation_split=0.2, callbacks
=[tensorboard])
40                 score = model.evaluate(test_input, test_output,
verbose=2)
41                 end = time.time()
42                 elapsed = end - start
43                 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))
44                 results.append((conv2d_count, dense_size, dropout,
score[0], score[1], elapsed))

```

▪ Penjelasan:

Baris 1: Import atau memasukkan modul time

Baris 2: Variabel result berisikan array kosong

Baris 3: Menggunakan convolution 2D yang berarti memiliki 1 atau 2 layer

Baris 4: Mendefinisikan dense size dengan ukuran 128, 256, 512, 1024, 2048

Baris 5: Mendefinsikan drop out dengan 0, 25%, 50%, dan 75%

Baris 6: Melakukan pemodelan Sequential

Baris 7: Jika ini adalah layer pertama, akan memasukkan bentuk input.

Baris 8: Kalau tidak kita hanya akan menambahkan layer.

Baris 9: Kemudian, setelah menambahkan layer konvolusi, lakukan hal yang sama dengan max pooling.

Baris 10: Lalu, ratakan atau flatten dan menambahkan dense size ukuran apa pun yang berasal dari dense size. Dimana akan selalu menggunakan algoritma tanh

Baris 11: Jika dropout digunakan, akan menambahkan layer dropout. Dropout ini berarti misalnya 50%, bahwa setiap kali memperbarui bobot setelah setiap batch, ada peluang 50% untuk setiap bobot yang tidak akan diperbarui

Baris 12: Menempatkan di antara dua lapisan padat untuk dihindarkan dari melindunginya dari overfitting. Lapisan terakhir akan selalu menjadi jumlah kelas karena itu harus, dan menggunakan softmax. Itu dikompilasi dengan cara yang sama.

Baris 13: Atur direktori log yang berbeda untuk TensorBoard sehingga dapat membedakan konfigurasi yang berbeda.

Baris 14: Variabel start akan memanggil modul time

Baris 15: Melakukan fit atau compile

Baris 16: Melakukan scoring dengan evaluate yang akan menampilkan data loss dan accuracy dari model

Baris 17: 'End' merupakan variabel untuk melihat waktu akhir pada saat pemodelan berhasil dilakukan.

Baris 18: Menampilkan hasil dari skrip diatas

- Hasil output:

```
....  
....      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, dense_size, dropout, score[0],  
score[1], elapsed))  
....      results.append((conv2d_count, dense_size, dropout, score[0],  
score[1], elapsed))  
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.13, Accuracy: 0.74, Time: 1540  
sec  
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.93, Accuracy: 0.76, Time: 1579  
sec  
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.81, Accuracy: 0.77, Time: 1599  
sec  
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.82, Accuracy: 0.77, Time: 1627
```

7.2.2.14 Soal No. 14 Jelaskan arti dari setiap baris kode program pada blok # In[14] dan hasil luarnya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model = Sequential()
9 model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
10                 input_shape=np.shape(train_input[0])))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12 model.add(Conv2D(32, (3, 3), activation='relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14 model.add(Flatten())
15 model.add(Dense(128, activation='tanh'))
16 model.add(Dropout(0.5))
17 model.compile(loss='categorical_crossentropy', optimizer='adam',
18                 metrics=['accuracy'])
18 print(model.summary())

```

- Penjelasan:

Baris 1: Melakukan pemodelan Sequential

Baris 2: Menambahkan Convolutio 2D dengan dmensi 32, dan ukuran matriks 3x3 dengan function aktivasi yang digunakan yaitu relu dan menampilkan input shape

Baris 3: Melakukan Max Pooling 2D dengan ukuran matriks 2x2

Baris 4: Melakukan Convolusi lagi dengan kriteria yang sama tanpa menambahkan input, ini dilakukan untuk mendapatkan data yang terbaik

Baris 5: Flatten digunakan untuk meratakan inputan atau masukkan data

Baris 6: Menambahkan dense input sebanyak 128 neuron dengan menggunakan function aktivasi tanh.

Baris 7: Dropout sebanyak 50% untuk menghindari overfitting

Baris 8: Menambahkan dense pada model untuk output dimana layerini akan menjadi jumlah dari class yang ada.

Baris 9: Mengcompile model yang didefinisikan diatas

Baris 10: Menampilkan ringkasan dari pemodelan yang dilakukan

- Hasil output:

```
In [14]: model = Sequential()
...: model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
input_shape=np.shape(train_input[0])))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Conv2D(32, (3, 3), activation='relu'))
...: model.add(MaxPooling2D(pool_size=(2, 2)))
...: model.add(Flatten())
...: model.add(Dense(128, activation='tanh'))
...: model.add(Dropout(0.5))
...: model.add(Dense(num_classes, activation='softmax'))
...: model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
...: print(model.summary())

Layer (type)                 Output Shape              Param #
=====conv2d_3 (Conv2D)        (None, 30, 30, 32)      896
=====max_pooling2d_3 (MaxPooling2D) (None, 15, 15, 32)    0
=====conv2d_4 (Conv2D)        (None, 13, 13, 32)      9248
=====max_pooling2d_4 (MaxPooling2D) (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 7.25 14

7.2.2.15 Soal No. 15 Jelaskan arti dari setiap baris kode program pada blok # In[15] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.fit(np.concatenate((train_input, test_input)),
9           np.concatenate((train_output, test_output)),
10          batch_size=32, epochs=10, verbose=2)

```

- Penjelasan:

Baris 1: Melakukan fit dengan join data train dan test agar dapat dilakukan pelatihan untuk jaringan pada semua data yang dimiliki.

- Hasil output:

```

In [26]: model.fit(np.concatenate((train_input, test_input)),
...:           np.concatenate((train_output, test_output)),
...:           batch_size=32, epochs=10, verbose=2)
Epoch 1/10
- 247s - loss: 1.7949 - acc: 0.5843
Epoch 2/10
- 236s - loss: 1.0797 - acc: 0.7064
Epoch 3/10
- 235s - loss: 0.9648 - acc: 0.7308
Epoch 4/10
- 237s - loss: 0.9060 - acc: 0.7434
Epoch 5/10
- 237s - loss: 0.8671 - acc: 0.7519
Epoch 6/10
- 236s - loss: 0.8362 - acc: 0.7573
Epoch 7/10
- 238s - loss: 0.8137 - acc: 0.7631
Epoch 8/10
- 239s - loss: 0.7980 - acc: 0.7652
Epoch 9/10
- 239s - loss: 0.7831 - acc: 0.7692
Epoch 10/10
- 239s - loss: 0.7695 - acc: 0.7724
Out[26]: <keras.callbacks.History at 0x14c1941f5f8>

```

Gambar 7.26 15

7.2.2.16 Soal No. 16 Jelaskan arti dari setiap baris kode program pada blok # In[16] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """

```

```

3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 model.save("mathsymbols.model")

```

- Penjelasan:

Baris 1: Menyimpan model yang telah di latih dengan nama mathsymbols.model

- Hasil output:

```
In [22]: model.save("mathsymbols.model")
```

Gambar 7.27 16

7.2.2.17 Soal No. 17 Jelaskan arti dari setiap baris kode program pada blok # In[17] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:50:12 2020
4
5 @author: User
6 """
7
8 np.save('classes.npy', label_encoder.classes_)

```

- Penjelasan:

Baris 1: Menyimpan label enkoder (untuk membalikkan one-hot encoder) dengan nama classes.npy

- Hasil output:

```
In [23]: np.save('classes.npy', label_encoder.classes_)
```

Gambar 7.28 17

7.2.2.18 Soal No. 18 Jelaskan arti dari setiap baris kode program pada blok # In[18] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:35 2020
4

```

```

5 @author: User
6 """
7
8 import keras.models
9 model2 = keras.models.load_model("mathsymbols.model")
10 print(model2.summary())

```

- Penjelasan:

Baris 1: Memasukkan atau mengimport models dari librari Keras

Baris 2: Variabel model2 akan memanggil model yang telah disave sebelumnya

Baris 3: Menampilkan ringkasan dari hasil pemodelan

- Hasil output:

```

In [24]: import keras.models
...: model2 = keras.models.load_model("mathsymbols.model")
...: print(model2.summary())

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 7.29 18

7.2.2.19 Soal No. 19 Jelaskan arti dari setiap baris kode program pada blok # In[19] dan hasil luarnya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:53:36 2020
4
5 @author: User
6 """
7

```

```

8 label_encoder2 = LabelEncoder()
9 label_encoder2.classes_ = np.load('classes.npy')
10
11 def predict(img_path):
12     newimg = keras.preprocessing.image.img_to_array(pil_image.
13         open(img_path))
14     newimg /= 255.0
15
16     # do the prediction
17     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
18
19     # figure out which output neuron had the highest score, and
20     # reverse the one-hot encoding
21     inverted = label_encoder2.inverse_transform([np.argmax(
22         prediction)]) # argmax finds highest-scoring output
23     print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
24         max(prediction)))

```

▪ Penjelasan:

Baris 1: Memanggil fungsi LabelEncoder

Baris 2: Variabel label encoder akan memanggil class yang disimpan sebelumnya.

Baris 3: Function Predict akan mengubah gambar kedalam bentuk array

Baris 4: Variabel prediction akan melakukan prediksi untuk model2 dengan reshape variabel newimg dengan bentukarray 4D.

Baris 5: Variabel inverted akan mencari nilai tertinggi output dari hasil prediksi tadi

▪ Hasil output:

```

In [25]: 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 7.30 19

7.2.2.20 Soal No. 20 Jelaskan arti dari setiap baris kode program pada blok # In[20] dan hasil luarannya.

- Code:

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Apr 13 23:54:14 2020
4
5 @author: User
6 """
7
8 predict("HASYv2/hasy-data/v2-00010.png")
9
10 predict("HASYv2/hasy-data/v2-00500.png")
11
12 predict("HASYv2/hasy-data/v2-00700.png")

```

- Penjelasan:

Baris 1: Melakukan prediksi dari pelatihan dari gambar v2-00010.png

Baris 2: Melakukan prediksi dari pelatihan dari gambar v2-00500.png

Baris 3: Melakukan prediksi dari pelatihan dari gambar v2-00700.png

- Hasil output:

```

In [26]: predict("HASYv2/hasy-data/v2-00010.png")
...
...: predict("HASYv2/hasy-data/v2-00500.png")
...
...: predict("HASYv2/hasy-data/v2-00700.png")
Prediction: \pitchfork, confidence: 0.00
Prediction: \copyright, confidence: 0.00
Prediction: J, confidence: 0.00

```

Gambar 7.31 20

7.2.3 Penanganan Error

7.2.3.1 Penanganan Error

- Screenshoot:

```

Traceback (most recent call last):
File "<ipython-input-3-fd9abfd31556>", line 1, in <module>
    model.fit(np.concatenate((train_input, test_input)),
NameError: name 'model' is not defined

```

Gambar 7.32 Error

- Code Error:

NameError: name 'model' is not defined

- Penanganan Error:

Berdasarkan error maka penyelesaiannya ialah melakukan pendefinisian variabel model sehingga code dapat dijalankan. Lakukan running pada code yang berada diatasnya dimana mendefinisikan variabel model.

DAFTAR PUSTAKA

- [1] R. Awangga, “Sampeu: Servicing web map tile service over web map service to increase computation performance,” in *IOP Conference Series: Earth and Environmental Science*, vol. 145, no. 1. IOP Publishing, 2018, p. 012057.

Index

disruptif, [xxiii](#)
modern, [xxiii](#)