CERDAS MENGUASAI GIT

CERDAS MENGUASAI GIT
Dalam 24 Jam

Rolly M. Awangga
Informatics Research Center

Kreatif
Industri
Nusantara

Kreatif Industri Nusantara

Penulis:
Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:
M. Yusril Helmi Setyawan

Penyunting:
Syafrial Fachrie Pane
Khaera Tunnisa
Diana Asri Wijayanti

Desain sampul dan Tata letak:
Deza Martha Akbar

Penerbit:
Kreatif Industri Nusantara

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

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

Cetakan Pertama, 2019

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

DAFTAR ISI

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini diciptakan bagi yang awam dengan git sekalipun.

R. M. AWANGGA

Bandung, Jawa Barat
Februari, 2019

ACKNOWLEDGMENTS

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

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

R. M. A.

ACRONYMS

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

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

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

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

SYMBOLS

A Amplitude

& Propositional logic symbol

a Filter Coefficient

$\mathcal{B}$ Number of Beats

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

INTRODUCTION

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

Informatics Research Center
Bandung, Jawa Barat, Indonesia

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

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

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

BAB 1

CHAPTER 1

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

BAB 2

CHAPTER 2

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

BAB 3

CHAPTER 3

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

BAB 4

CHAPTER 4

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

BAB 5

CHAPTER 5

5.1 1174069 - Fanny Shafira Damayanti

5.1.1 Teori

1. Jelaskan kenapa kata-kata harus di lakukan vektorisasi. Dilengkapi dengan ilustrasi atau gambar.
   Kata kata harus dilakukan vektorisasi dikarenakan atau bertujuan utk mengukur nilai kemunculan suatu kata yang serupa dari sebuah kalimat sehingga kata-kata tersebut dapat di prediksi kemunculanya. atau juga di buatnya vektorisasi data digunakan untuk memprediksi bobot dari suatu kata misalkan ayam dan kucing sama-sama hewan maka akan dibuat prediksi apakah kata tersebut akan muncul pada kalimat yang kira-kira memiliki bobot yang sama.

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

Gambar 5.1 Teori 1

2. Jelaskan mengapa dimensi dari vektor dataset google bisa sampai 300. Dilengkapi-dengan ilustrasi atau gambar.
   Dimensi dataset dari google bisa mencapai 300 karena dimensi dari vektor tersebut digunakan untuk membandingkan bobot dari setiap kata, misalkan terdapat kata dog dan cat pada dataset google tersebut setiap kata tersebut di buat dimensi vektor 300 untuk kata dog dan 300 dimensi vektor juga untuk kata cat kemudian kata tersebt di bandingkan bobot kesamaan katanya maka akan muncul akurasi sekitar 70 persen kesamaan bobot dikarenakan kata dog dan cat sama sama di gunakan untuk hewan priharaan.

Gambar 5.2 Teori 2

3. Jelaskan konsep vektorisasi untuk kata.dilengkapi dengan ilustrasi atau gambar.
   Vektorisasi untuk kata untuk mengetahui kata tengah dari suatau kalimat atau kata utama atau objek utama pada suatau 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 yany terdapat pada dokumen tersebut.

Gambar 5.4 Teori 4

5. Jelaskan apa mean dan standar deviasi,dilengkapi dengan ilustrasi atau gambar.
   mean merupakan petunjuk terhadap kata-kata yang di olah jika kata kata itu akurasinya tinggi berarti kata tersebut sering muncul begitu juga sebaliknya, sedangkan setandar defiation merupakan standar untuk menimbang kesalahan. sehingga kesalahan tersebut di anggap wajar misarkan kita memperkirakan kedalaman dari dataset merupakan 2 atau 3 tapi pada kenyataanya merupakan 5 itu merupakan kesalahan tapi masih bisa dianggap wajar karna masih mendekati perkiraan awal.

Gambar 5.5 Teori 5

6. Jelaskan apa itu skip-gram,dilengkapi dengan ilustrasi atau gambar.
   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. Cobalah dataset google, dan jelaskan vektor dari kata love, faith, fall, sick, clear,shine, bag, car, wash, motor, cycle dan cobalah untuk melakukan perbandingan similirati dari masing-masing kata tersebut.

   - berikut merupakan code import gensim digunakan untuk membuat data model atau rangcangan data yang akan di buat. selanjutnya dibuat variabel gmodel yang berisi data vektor negativ. selanjutnya data tersebut di load agar data tersebut dapat di tampilkan dan di olah.

```
In [1]: import gensim
        gmodel = gensim.models.KeyedVectors.load_word2vec_format('GoogleNews-vectors-negative300.bin', binary = True)
```

Gambar 5.7 Praktek 1

   - berikut merupakan hasil lpengolahan kata love pada data google yang di load tadi. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [3]: gmodel['love']
Out[3]: array([ 0.18902734, -0.15234375,  0.02587891,  0.10593906, -0.16503906,
                0.06689453,  0.29296875, -0.26367188, -0.140625  ,  0.20117188,
               -0.03124512, -0.08203125, -0.02770996, -0.04394531, -0.23151156,
                0.16992188,  0.12890625,  0.15722656,  0.00756836, -0.06082422,
               -0.03027422,  0.07958984,  0.22040219, -0.14355469,  0.10770875,
               -0.03515625,  0.05517578,  0.10069559,  0.11181641, -0.16308594,
               -0.11181641,  0.13964844,  0.05156296,  0.12792969,  0.15429688,
                0.07714844, -0.20171875,  0.08642578, -0.02514648,  0.13138438,
                0.10852344, -0.20996094,  0.07080078,  0.02008993, -0.18044531,
               -0.10253906,  0.12304688,  0.04711914,  0.02209473,  0.05834961,
               -0.10908210,  0.14944406, -0.10651550,  0.01556396,  0.08984375,
                0.11230469, -0.04370117, -0.11376953, -0.0037384 , -0.01818848,
                0.24316406,  0.08447266, -0.07080078,  0.18096406,  0.03515625,
               -0.09667969, -0.21972656, -0.00328064, -0.02168242,  0.18457031,
                0.20515625, -0.0859375 , -0.11181641,  0.0213022 , -0.30664062,
               -0.09228516, -0.18945312,  0.01513672,  0.18554688,  0.34375
               -0.16964688,  0.22558594,  0.06748634, -0.2265625 , -0.24692188,
                0.08251953, -0.34765682,  0.15390625,  0.25953125,  0.09218828,
               -0.12109375,  0.07080078, -0.09375   ,  0.14453125,  0.12109375,
               -0.11767578,  0.19726562, -0.01660767, -0.18408391, -0.1640625 ,
               -0.13720562,  0.19824219,  0.09521484,  0.08501523,  0.12597656,
                0.00073624, -0.0402832 , -0.03063965,  0.01627535, -0.1640625 ,
               -0.22167969,  0.171875  ,  0.12011719, -0.01065332,  0.4453125 ,
                0.06049141,  0.09312017, -0.1640625 , -0.01397188,  0.18945312,
                0.05566406, -0.09004883, -0.01422119,  0.15917969,  0.07421875,
               -0.16640625, -0.0534668 , -0.02355957, -0.16992188,  0.0625    ,
               -0.140625  ,  0.13183594, -0.12792969,  0.12060547,  0.05883789,
```

Gambar 5.8 Praktek 1.1

   - berikut merupakan hasil lpengolahan kata faith pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [4]: gmodel['faith']
Out[4]: array([ 0.20367188, -0.04158591,  0.1955125 ,  0.13476562, -0.14644438,
                0.11962891,  0.04345703,  0.10351562,  0.12207031,  0.13476562,
                0.06640625,  0.18945312,  0.16601562,  0.26376688, -0.27184438,
                0.3203125 ,  0.10669219,  0.36132812, -0.1951125 , -0.18164062,
                0.13132031, -0.10839844,  0.10253906, -0.01367188,  0.23144531,
               -0.05957031,  0.22948219, -0.00604246,  0.20171875,  0.10302734,
                0.1328125 ,  0.23404375,  0.01135254,  0.02111328,  0.18554688,
                0.04125977,  0.12011719,  0.17480469, -0.22167969, -0.13476562,
                0.3125    ,  0.06640625, -0.17675781,  0.01708984, -0.1448625 ,
               -0.02819024,  0.01257324, -0.09521484, -0.10866406, -0.108625 ,
               -0.02216301,  0.16308594, -0.11816394, -0.00007812,  0.13403538,
                0.27539062, -0.20605469,  0.10351562,  0.20214844, -0.1875    ,
                0.16992188,  0.13574219,  0.13769531,  0.16308594, -0.03881836,
                0.13132812,  0.05688477,  0.12255859,  0.09814453, -0.04956055,
               -0.02231542, -0.04240047, -0.00203125,  0.10811625,  0.04158591,
               -0.16601562, -0.13671875,  0.05659141,  0.32617188,  0.08211953,
               -0.20800781,  0.04199219,  0.05834961, -0.27734375,  0.09138059,
               -0.17382812, -0.22460938,  0.03466797,  0.19824219, -0.08837891,
                0.18559375,  0.07324219,  0.1171875 , -0.13984375,  0.16798675,
                0.13574219, -0.30075125, -0.04659771,  0.00805859, -0.29296875,
                0.13224375,  0.02966309,  0.13281125,  0.20120312,  0.09375    ,
               -0.20605469, -0.00082031,  0.0534668 ,  0.05834961, -0.01222656,
               -0.22266875,  0.25585938,  0.04430298,  0.140625 ,  0.09182647,
                0.21582031,  0.0219748 ,  0.02929688,  0.20019531,  0.34960938,
                0.38640219,  0.01504918,  0.04077148,  0.32226562, -0.03125    ,
               -0.05686677,  0.10058047, -0.04785156, -0.15136719, -0.07714844,
               -0.45898438, -0.20692188, -0.328125 , -0.20996046,  0.38671875,
                0.0019678 ,  0.07373047, -0.01220703,  0.12460938,  0.14550781,
```

Gambar 5.9 Praktek 1.2

- berikut merupakan hasil lpengolahan kata fall pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [5]: gmodel['fall']
Out[5]: array([-0.04272461,  0.10742188, -0.09277344,  0.16894531, -0.1328125 ,
               -0.10093359,  0.04322188,  0.01904297,  0.14648438,  0.15039062,
               -0.00091406,  0.04492188,  0.0145874 ,  0.08095406, -0.19824219,
               -0.11035156,  0.01092529, -0.00300781, -0.0109209 , -0.1953125 ,
               -0.1015625 ,  0.13671875,  0.09228516, -0.12109375,  0.12695312,
                0.03417969,  0.2109375 ,  0.01977539,  0.125     ,  0.01544189,
               -0.20953125, -0.0098877 ,  0.07763672, -0.15527344, -0.03369155,
                0.04199219, -0.20982812, -0.10554688,  0.08896094,  0.02007402,
                0.13574219, -0.22558594,  0.33789062, -0.03564453, -0.10839844,
                0.19335938,  0.0540875 , -0.04956055,  0.1071875 , -0.03295898,
                0.10205078, -0.15136719, -0.00445557,  0.04003906,  0.27539062,
                0.00933594,  0.05834961,  0.01422119, -0.01397705,  0.05395508,
               -0.0255127 ,  0.06298828,  0.07080078, -0.07017188,  0.06542969,
                0.01672363, -0.04711914,  0.13628906, -0.00094375,  0.078125 ,
                0.2109375 ,  0.0612793 ,  0.08789062,  0.13622086,  0.13369375,
                0.06542969,  0.03125     ,  0.12988281,  0.02270508,  0.14550781,
               -0.06225586, -0.37695312, -0.05737305, -0.06296484,  0.08984375,
                0.00446068, -0.14160156, -0.04541016, -0.0703125 ,  0.06005859,
                0.20757812,  0.02001953,  0.12695312, -0.04882812,  0.10945312,
                0.03366707,  0.00639672,  0.1484375 ,  0.01708984, -0.06709062,
               -0.14941406, -0.02331542, -0.03955078, -0.10400391, -0.14160156,
                0.10261719, -0.03076172,  0.04589844, -0.2890625 , -0.03540039,
                0.12890625, -0.10595703,  0.17578125,  0.05609453,  0.34960938,
                0.04296875,  0.09863281, -0.00956641, -0.06298828,  0.12255859,
                0.0234375 , -0.00404541,  0.00607969, -0.04980844,  0.04092085,
                0.00007812, -0.00402176, -0.1640625 ,  0.03271484,  0.0703125 ,
               -0.07958984, -0.12890625, -0.01037903, -0.17773438,  0.01235946,
               -0.20019531,  0.00886719, -0.18847656, -0.23828125,  0.02758789,
               -0.14453125, -0.10058594, -0.0059375 ,  0.10395078, -0.00396729,
```

Gambar 5.10 Praktek 1.3

- berikut merupakan hasil lpengolahan kata sick pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [6]: gmodel['sick']
Out[6]: array([ 1.02637188e-01,  1.49434082e-01, -4.05273438e-02,  1.64062500e-01,
               -2.54766631e-01,  1.22265625e-01,  1.73828125e-01, -1.47460938e-01,
                1.05074219e-01,  5.44875000e-01,  1.66992188e-01, -1.08045312e-01,
                2.34304594e-01,  5.66796875e-01, -1.60015625e-01, -1.12304688e-01,
                1.26953125e-01,  1.26953125e-01,  1.26953125e-01,  1.26953125e-01,
                8.74023438e-02,  2.56347656e-01,  1.82373875e-01,  4.08666875e-01,
                1.70738036e-01, -5.91021289e-04,  8.88671875e-02, -1.95312500e-01,
                1.83544625e-01,  2.65625000e-01, -1.45507812e-01,  1.00495938e-01,
                9.42382812e-02, -3.12500000e-02,  1.90374609e-02, -6.39648438e-02,
                1.18652344e-01,  1.23046875e-01, -6.89027344e-02,  4.49375000e-01,
                9.13065938e-02, -1.12500000e-01,  1.84570312e-01, -1.51367188e-01,
                5.78031281e-02,  1.04980469e-01, -1.80945312e-01,  8.80078125e-02,
               -2.01171875e-01,  1.06203125e-02, -1.29882812e-01, -1.29576562e-01,
               -5.56640625e-02,  2.14453125e-01,  5.61523438e-02, -1.26127188e-02,
                7.12890625e-02,  4.37031375e-02,  2.05078125e-01,  5.71289062e-02,
                4.64726562e-02,  2.15820312e-01, -1.26953125e-01,  8.78906250e-02,
                2.40046875e-01, -6.54296875e-02, -2.02163730e-02,  1.53347750e-01,
               -3.57421875e-01,  3.02148125e-01, -2.00007812e-01, -5.95173994e-02,
                2.83982422e-02,  1.77828125e-01, -2.00007812e-01, -5.93261719e-02,
               -6.49414062e-02,  5.65795516e-01,  2.35400250e-01,  2.77467590e-01,
                1.54001906e-02,  1.56250000e-01, -2.03857422e-02,  2.35152500e-01,
                1.83544625e-01,  2.65625000e-01, -1.45507812e-01,  1.00495938e-01,
                2.03173875e-01, -2.03173875e-01, -1.50756836e-01,  2.03173875e-01,
               -1.10030044e-01, -4.21875000e-01,  2.22187969e-01,  2.46484375e-01,
                4.19921875e-01, -6.84765625e-02,  9.42382812e-02,  1.96289062e-01,
               -9.42382812e-02, -3.12500000e-02,  6.34765625e-02,  2.47002734e-02,
                1.03132812e-01,  1.55328125e-01,  1.73828125e-01,  2.01440625e-02,
```

Gambar 5.11 Praktek 1.4

- berikut merupakan hasil lpengolahan kata clear pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [7]: gmodel['clear']

Out[7]: array([-2.4414062e-04, -1.0205078e-01, -1.4941406e-01, -4.2480468e-02,
               -1.6796875e-01, -1.4648437e-01, 1.7675781e-01, 1.4648437e-01,
                2.2050781e-01, 9.7656250e-02, -2.0757812e-01, -1.2988281e-01,
                1.2451172e-01, 2.2332812e-01, -2.1367188e-01, 3.1005859e-02,
                2.0019531e-01, -4.7607422e-02, -6.0556770e-02, -1.2109756e-01,
                3.2226562e-02, 3.1445312e-01, -1.1181640e-01, 8.0078125e-02,
               -2.7587890e-02, -6.0424804e-03, -7.3784688e-02, -1.7285156e-01,
                9.6679687e-02, -4.0123900e-02, -1.7871094e-01, -1.4038085e-02,
                1.1230469e-01, 1.3671875e-01, -1.2304688e-01, -1.1035156e-01,
                1.9007446e-01, -1.1411165e-02, 1.1865234e-01, 3.3447265e-02,
                2.5585937e-01, -7.1289062e-02, 2.0320312e-01, -2.7539062e-01,
               -6.4506912e-02, -2.7353516e-02, -0.1796875e-02, -1.1865234e-01,
                3.4912109e-02, 7.3242187e-02, 2.1777344e-01, 5.0048828e-02,
                7.4707031e-02, -1.2548800e-01, -5.7373046e-02, -2.7441210e-02,
               -1.7725390e-02, -2.1367188e-01, -1.1279296e-01, -4.9804687e-02,
               -1.9238281e-01, 1.1181640e-01, -6.0048828e-02, -1.6601562e-01,
                1.5742578e-02, -1.5703512e-01, 5.8183594e-02, 1.5724565e-01,
               -1.5234375e-01, -1.6474922e-02, -2.1753906e-01, 5.5447265e-02,
                1.5527344e-01, 1.6601562e-01, -1.6602100e-01, 2.1494140e-02,
                2.0556250e-01, 2.6955129e-01, 5.3281250e-02, 2.4123046e-01,
                1.3046156e-02, 5.1269531e-02, 9.7662500e-02, 3.1445146e-02,
                1.2304688e-01, 1.2109375e-01, -1.2304688e-01, 1.1865234e-01,
               -5.2050156e-02, 2.8066408e-01, -2.0703125e-01, -1.6015625e-01,
               -2.0123046e-02, -5.0151372e-02, 0.6689922e-01, -1.3085937e-01,
                2.0181640e-02, 8.6656703e-03, 5.1406640e-01, -6.6640750e-02,
                1.0375976e-02, 1.1378906e-01, -1.0997324e-01, -4.1724804e-02,
               -1.1474604e-01, -1.4770687e-02, 9.7167968e-02, -1.6601210e-01,
                5.6306640e-02, -2.7618402e-03, -1.0546875e-01, -1.0302734e-01,
```

Gambar 5.12 Praktek 1.5

- berikut merupakan hasil lpengolahan kata shine pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [8]: gmodel['shine']

Out[8]: array([-0.12402344, 0.25976562, -0.15917969, -0.27734375, 0.30273438,
                0.09908950, 0.3025781, -0.22940219, -0.18359375, 0.3671875 ,
               -0.10302734, 0.13671875, 0.25390625, 0.07128906, 0.02539062,
                0.21777344, 0.24022438, 0.5234375 , 0.12304688, -0.13335938,
               -0.05883780, 0.0612793 , -0.01040918, 0.07017188, 0.05102539,
                0.20019531, 0.10005938, 0.00162506, -0.05029297, 0.14648438,
               -0.34765625, 0.02563477, -0.23925781, -0.04516602, -0.00479126,
               -0.24121094, -0.18945312, -0.15234375, -0.45493164, 0.01434326,
                0.390625 , -0.2189375 , 0.1484375 , -0.13183594, 0.24511719,
               -0.24023438, -0.36132812, -0.12792969, 0.10595703, 0.09961100,
               -0.0246582 , 0.32226562, 0.13376953, 0.18164062, 0.04931641,
               -0.10153908, -0.09283813, -0.20882812, -0.171875 , -0.18945312,
               -0.01367188, -0.20898438, -0.07061128, -0.0095375 , -0.05395508,
               -0.14257812, -0.140625 , 0.03027344, -0.14453125, 0.359375 ,
                0.16113281, 0.22265625, 0.265625 , -0.06147656, -0.03007617,
                0.04760742, 0.08837891, -0.04272461, 0.05908203, 0.07128906,
                0.0153775, -0.11621094, 0.07128906, 0.01485800, -0.18644531,
                0.08880710, 0.13523438, 0.09667969, -0.11083984, 0.16015625,
                0.3359375 , -0.1875 , 0.14550781, 0.04663087, 0.07617188,
               -0.09521484, 0.08447266, 0.08117188, 0.11230469, -0.33984375,
               -0.25390625, 0.05200195, 0.2753906, -0.00390418, -0.31054688,
               -0.22940219, 0.14941406, -0.1053125 , 0.00496094, -0.00753784,
                0.078125 , 0.05908203, 0.02355957, 0.06147656, 0.32617188,
               -0.08760234, 0.10058594, -0.11474609, -0.18164062, 0.13378906,
                0.1123046 , -0.00088105, 0.00601406, 0.03080594, 0.0300293 ,
               -0.03417969, -0.00830078, 0.14160156, -0.00610141, -0.13328125,
                0.00823975, -0.15234375, 0.19042969, 0.04030469, 0.25 ,
               -0.00171661, 0.04589844, -0.05102539, -0.04052734, -0.03491211,
```

Gambar 5.13 Praktek 1.6

- berikut merupakan hasil lpengolahan kata bag pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [9]: gmodel['bag']

Out[9]: array([-0.03515625, 0.15234375, -0.12402344, 0.13378906, -0.11328125,
               -0.0133667 , -0.16113281, 0.16048828, -0.06819938, 0.140625 ,
               -0.06005859, -0.3046875 , 0.20996094, -0.04345703, -0.2189375 ,
               -0.05957031, -0.05457711, 0.10253906, 0.19042969, -0.09423828,
                0.18847656, -0.07958984, -0.11035156, -0.07910156, 0.06347656,
               -0.15527344, -0.18945312, 0.11132812, 0.27539062, -0.00787109,
                0.01806641, 0.06030463, 0.2578125 , 0.0324707 , -0.24600375,
               -0.05641992, -0.01011184, 0.24121094, -0.21875 , 0.07568359,
               -0.00814453, -0.16113281, 0.16503906, -0.09521484, -0.16601562,
               -0.41796875, 0.0300293 , 0.19433594, 0.2890625 , 0.12695312,
               -0.10024219, -0.05117578, 0.04296875, -0.10107422, 0.07324219,
               -0.13378906, 0.265625 , -0.00466919, 0.19628906, -0.10193604,
                0.14941406, 0.1484375 , 0.09619141, 0.21777344, 0.08544922,
               -0.02031824, 0.02539062, 0.07759766, 0.23242188, -0.19628906,
                0.27539062, 0.09130859, 0.23730469, 0.09033203, -0.28515625,
                0.05932617, 0.06591797, -0.01794434, -0.00065111, -0.1796875 ,
                0.05615234, -0.12207031, -0.09082031, -0.05786133, -0.09375 ,
               -0.30273438, -0.06396484, -0.00744629, -0.17971094, 0.00544922,
               -0.20410156, 0.13378062, 0.00228882, -0.39453125, -0.14453125,
               -0.128125 , -0.12695312, 0.00544922, 0.15234375, 0.09082109,
               -0.1484375 , 0.05566406, 0.02344238, 0.07519531, -0.21484375,
               -0.15722656, 0.3359375 , -0.04736328, -0.00485084, -0.17235562,
                0.27929688, 0.05566406, -0.10058594, -0.08813708, -0.20703125,
                0.03291016, -0.14560781, -0.15017969, 0.16601906, 0.234375 ,
                0.03588867, 0.04296875, -0.25 , 0.1171875 , -0.07714844,
               -0.00621861, 0.125 , 0.08888716, 0.15527344, -0.02185059,
               -0.15234375, -0.12890625, -0.14765625, -0.13769531, -0.10164062,
```

Gambar 5.14 Praktek 1.7

- berikut merupakan hasil lpengolahan kata car pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [10]: gmodel['car']

Out[10]: array([ 0.13085938,  0.00042285,  0.03344727, -0.05883789,  0.04003906,
                -0.14157813,  0.04931641, -0.16094531,  0.20898438,  0.11962891,
                 0.18066406, -0.25        , -0.16608591, -0.10742188, -0.01979882,
                 0.05200195, -0.00216675,  0.06445312,  0.14453125, -0.04541016,
                 0.11181641, -0.01611328, -0.03088379,  0.00447266,  0.10210938,
                 0.04467773, -0.15527344,  0.25390625,  0.33984375,  0.00756836,
                -0.21850938, -0.01731388, -0.03291888,  0.16308594, -0.12597656,
                -0.09912109,  0.16502906,  0.06884766, -0.18945312,  0.02832031,
                -0.0534668 , -0.03963965,  0.11803904,  0.24121094, -0.234375  ,
                 0.12353516, -0.00294495,  0.1484375 ,  0.33203125,  0.05240023,
                -0.20019531,  0.17995112,  0.12255859,  0.11425781, -0.13957781,
                 0.10009766,  0.0030365 ,  0.28757812,  0.20117188,  0.03710938,
                 0.11802304, -0.00014453, -0.1125    ,  0.03515625,  0.02832031,
                 0.26171875, -0.06662578, -0.02258301, -0.05834961, -0.00787354,
                 0.11767578, -0.04296875, -0.17285156,  0.06394531, -0.23466275,
                 0.1640625 , -0.11474609, -0.06038273,  0.01196289, -0.24707031,
                 0.13142578, -0.04492188, -0.13425781,  0.22851562,  0.05847949,
                -0.16859902, -0.11181594,  0.12597656, -0.17480469,  0.02200047,
                -0.1015625 ,  0.00617671,  0.10791016, -0.2600275 , -0.109375  ,
                -0.09375   , -0.01022555, -0.20214844,  0.23144531, -0.05446539,
                -0.05141992, -0.20898438,  0.26757812,  0.27921688,  0.17089844,
                -0.17578125, -0.02770996, -0.20410156,  0.02392578,  0.03125   ,
                -0.25390625, -0.125     , -0.05493164, -0.17382812,  0.28515625,
                -0.23342188,  0.0234375 , -0.20117188, -0.13476562,  0.26367188,
                 0.00769043,  0.20507812, -0.01708984, -0.12988281,  0.04711914,
                 0.22070312,  0.02099609, -0.23168562, -0.02893066,  0.17285156,
                 0.04372461, -0.19824219, -0.04003906, -0.16992188,  0.10058594,
                -0.09126172,  0.15820312, -0.16569906, -0.06056688,  0.19413594,
                -0.07080078, -0.06884766, -0.00619141, -0.07226562,  0.04882812,
```

Gambar 5.15 Praktek 1.8

- berikut merupakan hasil lpengolahan kata wash pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [11]: gmodel['wash']

Out[11]: array([ 3.41601562e-01, 1.41601562e-01, -5.46875000e-02, 1.54785156e-01,
                -2.38281250e-01, 3.24218750e-01, -6.44726562e-02, -1.29882812e-01,
                 1.07910156e-01, 2.53906250e-01, 1.13525391e-02, -1.66992188e-01,
                -2.79541016e-02, 2.00007812e-01, -4.27246094e-02, 1.85468750e-01,
                -7.42187500e-02, 3.06687500e-01, 2.11914062e-01, -0.88671875e-02,
                 2.07578125e-01, 2.12890625e-01, 1.74609546e-02, 2.02941895e-03,
                 6.26082812e-02, 1.62109375e-01, 1.93359375e-01, 2.17285156e-02,
                -2.67020000e-03, -9.13085938e-02, 2.32812500e-01, 2.23632812e-01,
                -8.00781250e-02, -3.08059375e-02, 1.00007656e-01, -1.39648438e-01,
                 1.74804688e-01, 6.70710938e-02, 1.11328125e-01, 1.65820062e-01,
                -1.05468750e-01, 2.50712891e-02, 2.00195312e-01, -6.89027344e-02,
                -1.61718800e-01, -1.42056870e-01, 1.30856375e-01, -5.85371094e-02,
                 1.07910156e-01, 1.05468750e-01, 1.07910156e-01, 1.07910156e-01,
                 2.50423828e-02, 1.05859188e-01, 6.85546800e-02, 4.13348016e-02,
                -5.56445312e-02, 1.77734375e-01, 1.12870312e-01, 3.73693750e-02,
                -1.16610604e-01, 6.83593750e-02, -2.52685547e-02, -1.27929688e-01,
                 4.21875000e-01, 5.32226562e-02, -5.93578125e-01, 1.74804688e-01,
                 1.77001953e-02, -2.05078125e-02, 2.21679688e-01, 3.18359375e-01,
                 1.00390438e-01, -4.30297852e-03, -2.45117188e-01, -2.06984375e-01,
                 3.58800781e-02, 8.30078125e-02, 1.60945312e-01, 2.79541016e-02,
                 1.04980469e-01, -3.47656250e-01, -5.20019531e-02, 2.24609375e-01,
                 1.09977734e-02, 1.69921875e-01, -1.46484375e-01, 2.65625000e-01,
                 2.17285156e-02, 1.12304688e-02, -1.14257812e-01, 7.22656250e-02,
                 4.52110004e-02, 1.11694336e-02, 8.07547266e-04, -7.91015625e-02,
                -5.98144531e-02, -5.44433594e-02, 3.78466875e-01, 5.62500000e-01,
                -2.20562500e-01, -6.39550781e-02, 1.13769531e-01, -1.83496604e-02,
                -1.55320312e-01, -4.37500000e-01, 2.59765625e-01, -1.40414062e-01,
                 5.66802246e-02, 2.13867188e-01, -2.04662246e-01, -1.74609375e-01,
```

Gambar 5.16 Praktek 1.9

- berikut merupakan hasil lpengolahan kata motor pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [12]: gmodel['motor']

Out[12]: array([ 5.73730469e-02, 1.50390625e-03, -4.01425781e-02, -1.32812500e-01,
                -2.59765625e-01, -1.77734375e-01, 3.68652344e-02, -4.37500000e-01,
                 2.36375000e-02, 2.57812500e-01, 1.76804688e-01, 2.44140625e-02,
                -2.53953125e-01, -5.76172875e-02, 8.15429688e-02, 1.86767578e-02,
                -3.81907031e-02, 1.59203125e-01, -5.85937500e-02, -1.12304688e-01,
                 1.56250000e-01, -4.24804688e-02, -1.32812500e-01, 2.13193402e-02,
                 1.23046875e-01, 1.69921875e-01, -1.55273438e-01, 4.59394375e-01,
                 3.46734375e-01, 1.53128112e-01, -1.69921875e-01, -1.01074219e-01,
                -1.16328086e-02, 2.28115625e-01, 8.06437500e-02, -7.12890625e-02,
                 1.07910156e-01, 1.07910156e-01, 1.07910156e-01, 1.07910156e-01,
                -4.42773344e-02, -1.86640625e-02, 7.42187500e-01, 5.58593750e-01,
                -1.98593750e-01, 1.80859938e-01, 1.54472656e-02, 2.10937500e-02,
                 3.10062656e-02, -6.50024414e-01, 6.34765625e-02, 4.02812012e-02,
                -2.78320312e-02, 1.07421875e-02, 1.47460938e-01, 2.00761719e-02,
                -1.58390625e-01, -1.17695312e-01, 9.96093750e-02, 1.23906250e-01,
                -3.34726562e-02, -1.00032227e-02, 2.16843750e-01, -9.52648438e-02,
                -6.39648438e-02, 7.51953125e-02, 1.09604625e-01, 2.17773438e-01,
                -2.21679688e-01, 2.33398438e-01, 5.05371094e-02, -3.37890625e-01,
                 1.13320312e-01, -7.12890625e-02, -3.68652344e-02, 7.66601562e-02,
                -0.00781250e-02, -1.14257812e-01, -9.71679688e-02, -2.61718750e-01,
                 3.84765625e-01, -1.87500000e-01, -1.10551562e-01, 1.00593594e-01,
                 1.00390438e-01, 9.57031250e-02, -8.20312500e-02, 1.54296875e-01,
                -2.40234375e-01, 8.34960936e-02, 4.19921875e-02, -1.91958594e-02,
                 9.71679688e-02, 2.52685547e-01, 5.68075000e-02, -5.88378906e-02,
                 8.20312500e-02, 2.32031250e-01, 2.17446350e-02, 5.71289062e-02,
                 1.77734375e-01, -9.57031250e-02, 2.45117188e-01, 8.84078562e-02,
                 2.63671875e-01, -8.15429688e-02, 1.15976562e-01, 1.20840094e-02,
                 4.00390625e-01, 8.89140625e-02, 5.00781250e-01, -1.99219750e-01,
```

Gambar 5.17 Praktek 1.10

- berikut merupakan hasil lpengolahan kata cycle pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [13]: gmodel['cycle']

Out[13]: array([ 0.04541016,  0.21679688, -0.02700061,  0.12153516, -0.20703125,
                -0.1328125 ,  0.26307188, -0.12890625, -0.125     ,  0.15332031,
                -0.18203759, -0.15820312, -0.06176758,  0.23972656, -0.15820312,
                 0.02563477, -0.07568359, -0.0625    ,  0.04014258, -0.31054688,
                -0.13739000, -0.13669922, -0.3359375 ,  0.078125  ,  0.08447266,
                 0.07226562, -0.06445312,  0.05513678,  0.14941406,  0.13671875,
                 0.10302734,  0.02172852, -0.10601559,  0.02490234, -0.10546125,
                -0.05544922,  0.25462109, -0.04859062,  0.06347656, -0.09447266,
                 0.14453125, -0.08203125,  0.08203125,  0.09375   ,  0.07031250,
                 0.11425781,  0.20800781, -0.00307102, -0.07173191,  0.15039062,
                 0.10046406, -0.20515625,  0.04052734,  0.01800641,  0.00131116,
                -0.00872003,  0.03564453,  0.23682812,  0.09900938, -0.16843275 ,
                -0.06707109,  0.05957031, -0.05513678, -0.10820000,  0.2265625 ,
                 0.01173828, -0.07080078,  0.1484375 , -0.20214844, -0.03393555,
                 0.09863281, -0.02030574, -0.08780062, -0.07226562, -0.09423828,
                -0.17090844,  0.1484375 ,  0.18564875,  0.08443512,  0.03851404,
                -0.02636719, -0.03686523, -0.125     , -0.06707109,  0.14267812,
                 0.37109375, -0.15722656,  0.09528372,  0.34900938, -0.09091553,
                -0.03613281,  0.16846531, -0.02656445,  0.10791016, -0.12421875,
                -0.14355469,  0.03173828, -0.07421875,  0.34179688,  0.140625 ,
                 0.0043335 , -0.12890625, -0.14960938, -0.02929688, -0.19629990,
                -0.2578125 , -0.3671875 ,  0.01463134,  0.20703125, -0.09667969,
                -0.10551562, -0.31054688,  0.02844238,  0.10400391,  0.17773438,
                 0.02197267, -0.14453125,  0.09375   ,  0.12109375,  0.09033203,
                 0.03203947, -0.13476562,  0.15527344,  0.13121032,  0.09091308,
                 0.19900804,  0.01605766,  0.15185546,  0.11718846,  0.02539062,
                 0.09900328, -0.09005406,  0.07275391, -0.35548875, -0.02749532,
                ...                     ...                     ...                     ...                     ...
```

Gambar 5.18 Praktek 1.11

- berikut merupakan hasil dari similaritas kata kata yang di olah menjadi matrix tadi adapun persentase untuk perbandingan setiap katanya yaitu 9 persen untuk kata wash dan clear 7 persen untuk kata bag dan love 48 persen untuk kata motor dan car 12 persen untuk kata sick dan faith dan terakhir yaitu 6 persen untuk kata cycle dan shine.

```
In [15]: gmodel.similarity('wash', 'clear')
Out[15]: 0.09019176

In [16]: gmodel.similarity('bag', 'love')
Out[16]: 0.07536096

In [17]: gmodel.similarity('motor', 'car')
Out[17]: 0.4810173

In [18]: gmodel.similarity('sick', 'faith')
Out[18]: 0.123073205

In [19]: gmodel.similarity('cycle', 'shine')
Out[19]: 0.061617922
```

Gambar 5.19 Praktek 1.12

2. Jelaskan dengan kata dan ilustrasi fungsi dari extract words dan PermuteSentences

```
In [20]: import re
         def extract_words(sent):
             sent = sent.lower()
             sent = re.sub(r'<[^>]+>', ' ', sent) #hapus tag html
             sent = re.sub(r'(\w)\'(\w)', ' ', sent) #hapus petik satu
             sent = re.sub(r'\W', ' ', sent) #hapus tanda baca
             sent = re.sub(r'\s+', ' ', sent) #hapus spasi yang berurutan
             return sent.split()

In [21]: import random
         class PermuteSentences(object):
             def __init__(self, sents):
                 self.sents = sents

             def __iter__(self):
                 shuffled = list(self.sents)
                 random.shuffle(shuffled)
                 for sent in shuffled:
                     yield sent
```

Gambar 5.20 Praktek 2

3. Jelaskan fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktek pemakaiannya.

```
In [22]: from gensim.models.doc2vec import TaggedDocument
         from gensim.models import Doc2Vec

In [ ]:
```

Gambar 5.21 Praktek 3

4. Jelaskan dengan kata dan praktek cara menambahkan data training dari file yang dimasukkan kepada variabel dalam rangka melatih model doc2vac.

```
In [52]: import os
         sents_sentences = []

In [54]: for dirname in ["train/pos","train/neg","train/unsup","test/pos","test/neg"]:
             for fname in sorted(os.listdir("./aclImdb/"+dirname)):
                 if fname[-4:] == '.txt':
                     with open('./aclImdb/'+dirname+'/'+fname,encoding='UTF-8') as f:
                         sent = f.read()
                         words = extract_words(sent)
                         sents_sentences.append(TaggedDocument(words,[dirname+'/'+fname]))

In [55]: for dirname in ["test_unsup/pos", "test_unsup/neg"]:
             for fname in sorted(os.listdir(dirname)):
                 if fname[-4:] == '.txt':
                     with open('./'+dirname+'/'+fname,encoding='UTF-8') as f:
                         for i, sent in enumerate(f):
                             words = extract_words(sent)
                             sents_sentences.append(TaggedDocument(words,["%s/%s-%d" % (dirname, fname, i)]))

In [56]: with open("stanfordsentiment treebank/original_rt_snippets.txt", encoding='UTF-8') as f:
             for i, sent in enumerate(f):
                 words = extract_words(sent)
                 sents_sentences.append(TaggedDocument(words,["r%-%d" % i]))
```

Gambar 5.22 Praktek 4

5. Jelaskan dengan kata dan praktek kenapa harus dilakukan pengocokan dan pembersihan data.

```
In [77]: import re
         def extract_words(sent):
             sent = sent.lower()
             sent = re.sub(r'<[^>]>', ' ', sent) #hapus tag html
             sent = re.sub(r"\w+:\/\/\S+", " ", sent) #hapus pattern seta
             sent = re.sub(r"[^\w\d\'\s]+", " ", sent) #hapus tanda baca
             sent = re.sub(r"\s+", " ", sent) #hapus spasi yang berurutan
             return sent.split()

         import random
         class PermuteSentences(object):
             def __init__(self, sents):
                 self.sents = sents

             def __iter__(self):
                 shuffled = list(self.sents)
                 random.choice(shuffled)
                 for sent in shuffled:
                     yield sent

         auto-PermuteSentences(sents_sentences)
         model = Doc2Vec(dm=0, dbow=0, hs=1,size=50)
         model.delete_temporary_training_data(keep_inference=True)
```

Gambar 5.23 Praktek 5

6. Jelaskan dengan kata dan praktek kenapa model harus di save dan kenapa temporari training harus dihapus.

```
In [78]: model.delete_temporary_training_data(keep_inference=True)
         model.save('hasil.d2v')
```

Gambar 5.24 Praktek 6

7. Jalankan dengan kta dan praktek maksud dari infer code.

```
In [79]: model.infer_vector(extract_words("I will go home"))

Out[79]: array([-1.7533980e-01,  1.7054052e-01,  8.7283678e-02,  1.7998287e-02,
                 6.2981184e-01, -3.2546040e-02,  1.6391001e-02, -1.1790003e-02,
                 1.1714227e-02,  4.6473891e-02,  1.6009269e-02,  2.6200551e-02,
                -2.5993909e-02,  7.7027005e-02, -1.5409571e-02,  1.3232007e-01,
                -4.3078101e-02,  4.0155737e-02, -5.1151481e-02, -7.2681673e-02,
                -5.1809944e-02, -2.4982825e-02,  1.1855837e-01,  7.9185477e-02,
                6.3015221e-03,  6.1555382e-02, -2.2586747e-01,  1.3894421e-01,
                1.6294651e-02, -6.1561765e-02,  1.1732188e-01, -7.6827034e-02,
                -1.4881554e-01,  4.0382754e-02, -1.2432394e-01, -3.1883963e-02,
                6.4639184e-02,  7.1162600e-02, -6.4144008e-02, -5.9015360e-02,
                1.0170000e-04,  1.0674181e-02,  6.5018474e-02,  7.7513527e-03,
                5.5747358e-02, -1.3002630e-02, -5.5843181e-02,  4.9702202e-02,
                -1.1787591e-01,  9.5305327e-02], dtype=float32)
```

Gambar 5.25 Praktek 7

8. Jelaskan dengan praktek dan kata maksud dari cosine similarity.

```
In [83]: from sklearn.metrics.pairwise import cosine_similarity
         cosine_similarity(
             [model.infer_vector(extract_words("she going to school, after wash hand"))],
             [model.infer_vector(extract_words("Services sucks2."))])

Out[83]: array([[0.04744039]], dtype=float32)

In [84]: from sklearn.metrics.pairwise import cosine_similarity
         cosine_similarity(
             [model.infer_vector(extract_words("Dia pergi ke sekolah tadi siang"))],
             [model.infer_vector(extract_words("Services sucks2."))])

Out[84]: array([[0.20954691]], dtype=float32)
```

Gambar 5.26 Praktek 8

9. Jelaskan dengan praktek score dari cross validation masing-masing metode.

```
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()

scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
print((np.mean(scores), np.std(scores)))

scores = cross_val_score(clfrf, sentvecs, sentiments, cv=5)
print((np.mean(scores), np.std(scores)))

# bag-of-words comparison
from sklearn.pipeline import make_pipeline
from sklearn.feature_extraction.text import CountVectorizer,
    TfidfTransformer
pipeline = make_pipeline(CountVectorizer(), TfidfTransformer(),
    RandomForestClassifier())
scores = cross_val_score(pipeline, sentences, sentiments, cv=5)
print((np.mean(scores), np.std(scores)))
```

5.1.3 Penangan Error

1. SS Error

```
  File "D:/Kuliah/Semester 6/Kecerdasan Buatan/Bahan/src/1174027/5/1174027.py",
line 16, in <module>
    scores = cross_val_score(clf, sentvecs, sentiments, cv=5)

NameError: name 'sentvecs' is not defined
```

Gambar 5.27 Name Error

2. Jenis error
   - Name Error
3. Cara Penanganan
   Dengan Mengecek Kembali Baris Code Yang Dibuat

5.1.4 Bukti Tidak Melakukan Plagiat

Gambar 5.28 Bukti Tidak Melakukan Plagiat

5.1.5 Link Youtube

https://youtu.be/NKurTLe8MpI

5.2 1174086 - Tia Nur Candida

5.2.1 Teori

1. Jelaskan kenapa kata-kata harus di lakukan vektorisasi. Dilengkapi dengan ilustrasi atau gambar.
   Karena untuk mengukur nilai kemunculan sebuah atau suatu kata yang serupa dari sebuah kalimat sehingga kata - kata tersebut dapat diprediksi kemunculannya. Dibuatnya vektorisasi data digunakan untuk memprediksi bobot dari suatu kata misalkan bunga dan daun keduanya sama - sama tumbuhan maka akan dibuat prediksi apakah kata tersebut akan muncul pada kalimat yang kira-kira memiliki bobot yang sama.

Gambar 5.29 Teori 1

2. Jelaskan mengapa dimensi dari vektor dataset google bisa sampai 300. Dilengkapi-dengan ilustrasi atau gambar.
   Karena dimensi dari vektor tersebut digunakan untuk membandingkan bobot dari setiap kata, misalkan terdapat kata dog dan cat pada dataset google tersebut setiap kata tersebut di buat dimensi vektor 300 untuk kata dog dan 300 dimensi vektor juga untuk kata cat kemudian kata tersebt di bandingkan bobot kesamaan katanya maka akan muncul akurasi sekitar 70 persen kesamaan bobot dikarenakan kata dog dan cat sama sama di gunakan untuk hewan peliharaan.

Gambar 5.30 Teori 2

3. Jelaskan konsep vektorisasi untuk kata.dilengkapi dengan ilustrasi atau gambar.
   Untuk mengetahui kata tengah dari suatau kalimat atau kata utama atau objek utama pada suatau kalimat contoh (Jangan lupa subscribe channel saya ya sekian terimakasih) 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 5.31 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 5.32 Teori 4

5. Jelaskan apa mean dan standar deviasi,dilengkapi dengan ilustrasi atau gambar.
   Mean merupakan petunjuk terhadap kata-kata yang di olah jika kata kata itu akurasinya tinggi berarti kata tersebut sering muncul begitu juga sebaliknya, sedangkan setandar defiation merupakan standar untuk menimbang kesalahan. sehingga kesalahan tersebut di anggap wajar misarkan kita memperkirakan kedalaman dari dataset merupakan 2 atau 3 tapi pada kenyataanya merupakan 5 itu merupakan kesalahan tapi masih bisa dianggap wajar karna masih mendekati perkiraan awal.

Gambar 5.33 Teori 5

6. Jelaskan apa itu skip-gram,dilengkapi dengan ilustrasi atau gambar.
   Skip-Gram adalah kebalikan dari konsep vektorisasi untuk kata dimana kata tengah menjadi acuan terhadap kata kata pelengkap dalam suatu kalimat.

Gambar 5.34 Teori 6

5.2.2 Praktek

1. Cobalah dataset google, dan jelaskan vektor dari kata love, faith, fall, sick, clear,shine, bag, car, wash, motor, cycle dan cobalah untuk melakukan perbandingan similirati dari masing-masing kata tersebut.

   - berikut merupakan code import gensim digunakan untuk membuat data model atau rangcangan data yang akan di buat. selanjutnya dibuat variabel gmodel

yang berisi data vektor negativ. selanjutnya data tersebut di load agar data tersebut dapat di tampilkan dan di olah.

```
In [2]: import gensim
        google = gensim.models.KeyedVectors.load_word2vec_format('GoogleNews-vectors-negative300.bin', binary = True)
```

Gambar 5.35 Praktek 1

- berikut merupakan hasil lpengolahan kata love pada data google yang di load tadi. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [3]: google['love']

Out[3]: array([ 0.10302734, -0.15234375,  0.02587891,  0.10503906, -0.16503906,
                0.06604453,  0.29296875, -0.26367188, -0.140625  ,  0.20117188,
               -0.02024412, -0.08203125, -0.02778906, -0.04394531, -0.23151156,
                0.16992188,  0.12304625,  0.15722656,  0.00756836, -0.46832422,
               -0.03027422,  0.07421594,  0.12646219, -0.14355469,  0.14746094,
                0.03318665,  0.09521594,  0.10595703,  0.12695312,  0.17578125,
               -0.11181641,  0.13964844,  0.09155156,  0.12792969,  0.15429688,
                0.07714844,  0.26171875,  0.00642578, -0.02514648,  0.13366438,
                0.18052344,  0.20390854,  0.07080078,  0.02008058, -0.10644531,
               -0.10529306,  0.12304688,  0.04711914,  0.01209473,  0.05614063,
               -0.16908228,  0.14041406, -0.10693359,  0.01556396,  0.08064375,
                0.11229492, -0.04170117, -0.11370955, -0.0037344 , -0.03318848,
                0.24316406,  0.08447266, -0.07080078,  0.10066406,  0.03515625,
               -0.00667969, -0.23972656, -0.00328064, -0.03198242,  0.18457031,
                0.20155625,  -0.0859375 ,  -0.11181641,  0.0213623 ,  -0.30664062,
               -0.09228516, -0.18945312,  0.03513672,  0.10534688,  0.34179 ,
               -0.10956688,  0.22558594,  0.06748236, -0.2265625 ,  -0.20492188,
                0.08251953, -0.18476562,  0.25390625,  0.20953125,  0.06238828,
               -0.00958252,  0.23632812, -0.17071094, -0.12451172, -0.17285156,
               -0.11767578,  0.19726562, -0.01660767, -0.10400391, -0.16406625 ,
               -0.19720562,  0.19824219,  0.09521484,  0.09501523,  0.12537656,
                0.00073624, -0.0402832 , -0.03063965,  0.03622635, -0.16406625 ,
               -0.22167969,  -0.171875 ,  0.12011719, -0.01965332,  0.4453125 ,
                0.06604541,  0.09521027, -0.1640625 , -0.01707188,  0.18945312,
                0.05566406, -0.09404883, -0.01422119,  0.15917969,  0.07421875,
               -0.16406625, -0.0614668 , -0.02355957, -0.10992188,  0.0625  ,
               -0.140625  ,  -0.13183594, -0.12792969,  0.12008547,  0.05883789,
```

Gambar 5.36 Praktek 1.1

- berikut merupakan hasil lpengolahan kata faith pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [4]: google['faith']

Out[4]: array([ 0.26367188, -0.04150391,  0.1953125 ,  0.13476562, -0.14404438,
                0.11962891,  0.06345703,  0.18955162,  0.12207031, -0.11425662,
                0.06640625,  0.18945312, -0.16601562,  0.21679688, -0.27148438,
                0.1328125 ,  0.18442219,  0.26512812, -0.1953125 ,  -0.18164062,
                0.15332031, -0.18039844,  0.18253906, -0.01367188,  0.23144531,
               -0.09957031,  -0.2304125 ,  0.09094248,  0.26171875,  0.18108724,
               -0.1328125 ,  0.21484375,  0.01135254,  0.02115156,  0.18554688,
                0.04125977,  0.12011719,  0.1748046 , -0.22167969, -0.11425662,
                0.3125  ,  0.06640625, -0.17675781, -0.01708984, -0.1640625 ,
               -0.03881024,  0.01227324, -0.09521484, -0.18066406, -0.140625  ,
               -0.02258301,  0.16308594, -0.13183594, -0.08007812,  0.13085938,
                0.27539062, -0.19695469,  0.19035162,  0.20214844, -0.1875  ,
                0.16992188,  0.13576219,  0.11769531,  0.16308594, -0.03881836,
               -0.11132812,  0.05688477,  0.12255859,  0.09814453, -0.04916406,
               -0.02331543, -0.04248047, -0.00203125,  0.10015625,  0.04158591,
               -0.16601562,  0.13671875,  0.09619141,  0.32617188,  0.08251953,
               -0.20800781,  0.04199219,  0.06181641, -0.27734375,  0.09138850,
               -0.17382812, -0.22460938,  0.05466797,  0.13824219, -0.08837891,
                0.18359375,  0.07324219,  0.1171875 , -0.33984375,  0.10796875,
               -0.1576219, -0.09078125,  0.09069791,  0.09985855, -0.23238075,
                0.15234375,  0.02966309,  0.13203125,  0.28320312,  0.09375  ,
               -0.24665069, -0.09082031,  0.0546668 ,  0.05814961, -0.03222656,
               -0.20196875,  0.25585938,  0.00438208,  0.140625  ,  0.05810547,
                0.21582031,  0.0191748 ,  0.02929688,  0.20019531,  0.14498038,
                0.18449219, -0.01948918,  0.04077148,  0.32226562, -0.1953125 ,
               -0.05688477,  0.18603847, -0.04793156, -0.15136719, -0.07734404,
               -0.41806638, -0.20492188, -0.320125 , -0.20996094,  0.38671875,
                0.4036678 ,  0.07737047, -0.01220703, -0.22460938,  0.14538701,
```

Gambar 5.37 Praktek 1.2

- berikut merupakan hasil lpengolahan kata fall pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [5]: gmodel['fall']

Out[5]: array([-0.04272461, 0.10742188, -0.09277344, 0.16034531, -0.1328125 ,
               -0.10093359, 0.04322289, 0.01904297, 0.14648438, 0.15039062,
               -0.00691406, 0.04492188, 0.0145874 , 0.08091406, -0.19024219,
               -0.11035156, 0.01092529, -0.06300781, -0.0109209 , -0.1353125 ,
               -0.1015625 , 0.13671875, 0.09228516, -0.12199375, 0.12695312,
               0.03417969, 0.2109375 , 0.01977539, 0.125 , 0.0154189,
               -0.20953125, -0.0098877 , -0.07763672, -0.15527344, -0.05393555,
               0.04199219, -0.20982812, -0.10554688, 0.08696094, -0.02007402,
               0.13574219, -0.22558594, 0.13759062, -0.03564453, -0.10639844,
               -0.13135938, 0.0540875 , -0.04956055, 0.1671875 , -0.03295898,
               0.10205078, -0.15136719, -0.00445557, 0.04069398, 0.27539062,
               -0.00933594, 0.05834961, 0.01422119, -0.01397705, 0.05395508,
               -0.0255127 , -0.06208828, 0.07080078, -0.07017188, 0.06542969,
               -0.01672363, -0.04711914, 0.13628906, -0.00094375, 0.078125 ,
               0.2109375 , 0.0612793 , 0.08789062, 0.13602906, 0.1137695,
               0.06542969, 0.03125 , 0.12988281, 0.02270508, 0.14550781,
               -0.06225586, -0.37695312, -0.05737305, -0.06396484, 0.08906375,
               0.08448608, -0.14160156, -0.04541016, -0.0703125 , 0.06005859,
               0.20757812, 0.03001953, -0.12695312, -0.04882812, 0.10945312,
               -0.03466797, 0.06638672, 0.1484375 , -0.01708984, -0.06709062,
               -0.14941406, -0.02131541, -0.03955078, -0.10400391, -0.14160156,
               -0.12016719, -0.03976172, 0.04589844, -0.2800625 , -0.03560039,
               0.12890625, -0.10916703, 0.17578125, 0.06608451, 0.14660938,
               0.04260875, 0.09863281, -0.00956641, -0.06298828, 0.12255859,
               0.0234375 , -0.00404441, 0.09667969, -0.04588844, 0.04956055,
               0.00007812, -0.00482170, -0.1640625 , -0.03271484, 0.0703125 ,
               -0.07958984, 0.12890625, -0.01037983, -0.17773438, 0.01203948,
               -0.20019531, 0.00886719, -0.18847656, -0.23828125, 0.02758789,
               -0.1453125, 0.10958594, -0.0059375 , 0.10205078, -0.00396729,
```

Gambar 5.38 Praktek 1.3

- berikut merupakan hasil lpengolahan kata sick pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [6]: gmodel['sick']

Out[6]: array([ 1.02637188e-01, 1.49434062e-01, -4.85273438e-02, 1.64062500e-01,
               -2.54394531e-01, 1.22216125e-01, 1.73828125e-01, -1.47460938e-01,
               1.81074219e-01, 5.46875000e-02, 1.60931188e-01, -1.18945312e-01,
               1.07421875e-01, 1.07421875e-01, 1.07421875e-01, 1.07421875e-01,
               1.09953125e-01, 1.79687500e-01, 5.93040156e-05, 2.45117188e-01,
               8.74023438e-02, 2.56347656e-02, 3.43796875e-01, 4.98046875e-02,
               1.78710938e-01, -6.01821289e-04, 8.88671875e-02, -1.05121094e-01,
               1.01464062e-01, -2.05825000e-03, -1.45507812e-01, 1.00585938e-01,
               9.42382812e-02, -1.12500000e-02, 1.98974609e-02, -6.39648438e-02,
               1.10652344e-01, 1.23046875e-01, -6.03027344e-02, 4.68750000e-01,
               9.13085936e-02, -1.12500000e-01, 1.84570312e-01, -1.51367188e-01,
               5.78613281e-02, -1.84082031e-01, -1.68945312e-01, -8.04785156e-02,
               -2.01171875e-01, 1.06201172e-02, -1.20882812e-01, -1.23076562e-02,
               -5.56640625e-02, 1.34453125e-01, 5.61523438e-02, -1.28117188e-01,
               7.12890625e-02, 4.37617336e-02, 2.66870312e-01, 5.71289062e-02,
               8.44726562e-02, 2.15820312e-01, -1.26953125e-01, 8.78906250e-02,
               2.40966875e-01, 6.54296875e-02, -2.02554735e-02, 1.52343750e-01,
               -3.57421875e-01, 3.02124023e-02, -2.00007812e-01, -5.46173094e-02,
               2.03582422e-02, 1.73828125e-01, -2.00007812e-01, -5.93262734e-02,
               6.44434062e-02, 1.63276531e-02, 1.91406250e-01, 2.77343750e-01,
               -1.07421875e-01, -1.07421875e-01, -1.07421875e-01, -1.07421875e-01,
               -4.54100156e-02, -1.57578125e-01, -2.63085938e-01, 4.17480469e-02,
               2.01171875e-01, -2.01171875e-01, -1.50756836e-02, 2.03738750e-01,
               -1.18659844e-01, -4.21875000e-01, 2.22167969e-02, 1.44404375e-01,
               4.19921875e-01, -6.86876562e-02, 9.42382812e-02, -1.96289062e-01,
               -6.42382812e-02, -1.12500000e-02, 6.34765625e-02, 2.47802734e-02,
               -1.63332812e-01, -1.53320312e-01, 1.31835938e-01, -1.81640625e-01,
```

Gambar 5.39 Praktek 1.4

- berikut merupakan hasil lpengolahan kata clear pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [7]: gmodel['clear']

Out[7]: array([-2.44140625e-04, -1.02050781e-01, -1.49414062e-01, -4.24804688e-02,
               -1.67968750e-01, -1.46484375e-01, 1.76757812e-01, 1.46484375e-01,
               2.26562500e-01, 9.76562500e-02, 2.67578125e-01, -1.20882812e-01,
               1.34513730e-01, 2.23632812e-01, -2.13867188e-01, 3.10059594e-02,
               2.00195312e-01, -6.76607422e-02, -6.03027344e-02, -1.21007750e-01,
               2.22656250e-02, 3.14453125e-01, -1.11816406e-01, 8.00781250e-02,
               -2.75390604e-02, -6.04248047e-02, -7.37304688e-02, -1.72851562e-01,
               9.66796875e-02, -4.91332008e-03, -1.78710938e-01, -1.40508850e-03,
               7.66601562e-02, 1.07910156e-01, -1.10351562e-01, -8.34960938e-02,
               1.98974609e-02, -3.14532895e-03, 1.98925234e-02, 3.34472656e-02,
               -1.07421875e-01, -1.07421875e-01, -1.07421875e-01, -1.07421875e-01,
               4.60449219e-02, -7.72705078e-02, 2.37304688e-01, -2.56835938e-01,
               3.49121094e-02, 7.32421875e-02, 5.17773438e-02, 1.00439235e-02,
               7.47070312e-01, 1.25000000e-01, -5.77734375e-02, -2.74658203e-02,
               -1.17319200e-02, -2.13867188e-01, -1.12792969e-01, -4.98046875e-02,
               -1.91928222e-01, 1.53125000e-01, 5.00488281e-02, 1.64019925e-01,
               -1.57470703e-02, -1.57695312e-01, 3.00183594e-02, 1.57226562e-01,
               -1.52343750e-01, -1.64794922e-02, -2.27539062e-01, 1.34472656e-02,
               1.55273438e-01, 1.66839062e-01, -1.66992188e-01, 3.14941406e-02,
               2.05156250e-01, 2.66552160e-01, 3.38012500e-02, 2.41210938e-01,
               1.99160156e-02, 5.13695312e-02, 9.76562500e-02, 3.14941406e-02,
               2.51464844e-02, -2.05156250e-01, 1.24022656e-01, -6.69475562e-02,
               -5.20703156e-02, 2.06054688e-01, -2.67031250e-01, -1.66552500e-01,
               -1.07421875e-01, -1.07421875e-01, -1.07421875e-01, -1.07421875e-01,
               -2.08828906e-02, 8.60565703e-03, 5.59863594e-01, -6.05468750e-02,
               1.03750766e-02, 1.33789062e-01, 1.09073524e-01, -4.17344094e-02,
               -1.44766844e-01, -1.47700703e-02, 9.71679688e-02, -1.66992188e-01,
               -5.63964844e-02, -2.76184082e-03, -1.05468750e-01, -1.03027344e-01,
```

Gambar 5.40 Praktek 1.5

- berikut merupakan hasil lpengolahan kata shine pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [8]: gmodel['shine']
Out[8]: array([-0.12402344,  0.25976562, -0.15917969, -0.17734375,  0.30273438,
                0.09960938,  0.39257812, -0.22949219, -0.18359375,  0.3671875 ,
               -0.10302734,  0.13671875,  0.25390625,  0.07128906,  0.02539062,
                0.21777344,  0.24012436,  0.5234375 ,  0.12304688, -0.13555938,
               -0.05883789,  0.0612793 , -0.01040018,  0.07017188,  0.05102539,
                0.20019531,  0.10805938,  0.00162506, -0.05029297,  0.14648438,
               -0.34765625,  0.02563477, -0.23925781, -0.04516602, -0.00479126,
                0.24121094, -0.10945312, -0.15234375, -0.05493164,  0.01434326,
                0.598625  , -0.2109375 ,  0.1484375 , -0.13183594,  0.24511719,
               -0.24023438, -0.16112812, -0.12792969,  0.10595703,  0.09912109,
               -0.0246582 ,  0.32226562,  0.11376953,  0.10164062,  0.04031641,
               -0.10253906, -0.00238312, -0.28082812,  0.171875  , -0.18945312,
               -0.01367188, -0.20898438, -0.07061328, -0.0859375 ,  0.05395508,
                0.14257812, -0.140625  ,  0.03027344, -0.14453125,  0.359375  ,
                0.16113281,  0.22265625,  0.265625  , -0.06347656, -0.02007017,
                0.04760742,  0.08017891, -0.04272461,  0.05908203,  0.07128906,
                0.01519775, -0.11621094,  0.07128906,  0.01403809, -0.10644531,
                0.08880715,  0.13523438,  0.09667969, -0.11803984,  0.10015625,
                0.3359375 , -0.1875    ,  0.14550781,  0.08463867,  0.07617188,
               -0.09121484,  0.09447266,  0.28117188,  0.11218409, -0.33364375,
               -0.25390625,  0.06200195,  0.27539062, -0.00390488, -0.31054688,
               -0.22949219,  0.14941406, -0.1953125 ,  0.00496094, -0.00753784,
                0.078125  ,  0.05908203,  0.02355957,  0.06347656,  0.22017188,
               -0.00740234,  0.10058594, -0.11474609, -0.10164062,  0.13378906,
                0.11230469, -0.00080109,  0.08691406,  0.03808594,  0.0300293 ,
               -0.03417969, -0.00030078,  0.14160156, -0.09019141, -0.13328125,
                0.00812375, -0.15234375,  0.19042969,  0.04980469,  0.25    ,
               -0.00171661,  0.04598944, -0.01102539, -0.00452734, -0.03491211.
```

Gambar 5.41 Praktek 1.6

- berikut merupakan hasil lpengolahan kata bag pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [9]: gmodel['bag']
Out[9]: array([-0.03515625,  0.15234375, -0.12402344, -0.13378906, -0.11328125,
               -0.0313667 , -0.16113281, -0.16448834, -0.06035938, -0.140625  ,
               -0.06005859, -0.3046875 ,  0.20996094, -0.04345703, -0.2109375 ,
               -0.05957031, -0.05453711,  0.10253906,  0.19042969, -0.09421828,
                0.18847656, -0.07958984, -0.11035156, -0.07910156, -0.05347656,
                0.15527344, -0.10945312,  0.11132812,  0.27539062, -0.00378180,
                0.01806641, -0.06689453, -0.2578125 ,  0.0324707 , -0.24609375,
               -0.05681992,  0.01811184,  0.24121094, -0.21875   ,  0.07688599,
               -0.00814453, -0.16113281, -0.16503906, -0.09521484, -0.16601562,
               -0.41796875,  0.0300293 , -0.19433594,  0.2890625 , -0.12695312,
               -0.19824219, -0.05117578,  0.04296875, -0.10107422,  0.07324219,
               -0.13378906,  0.265625  , -0.00466919, -0.19628906, -0.10198844,
                0.14941406, -0.1484375 ,  0.09619141,  0.21777344,  0.08544922,
                0.02018824,  0.02539062, -0.07759766,  0.23242188,  0.19628906,
                0.27539062,  0.09130859,  0.23730469,  0.00833203, -0.28115625,
                0.05322217,  0.06591797, -0.0174444,  -0.00055313, -0.1796875 ,
                0.05615234, -0.12207031, -0.00863281, -0.05786133, -0.09375   ,
               -0.36273438, -0.06396484, -0.00746625, -0.17071094,  0.00544922,
               -0.20410156,  0.13789062,  0.00228882, -0.39453125, -0.14453125,
                0.328125  , -0.12695312, -0.00544922, -0.15234375,  0.07656250,
               -0.1484375 ,  0.05566406,  0.02344238,  0.07519531, -0.21484375,
               -0.15722656,  0.159375  , -0.04735328, -0.00405884, -0.13735562,
                0.27929688,  0.05566406, -0.10058594, -0.00811768, -0.20703125,
                0.03203888, -0.14550781, -0.15017969,  0.16580906,  0.234375  ,
                0.03588867,  0.04296875, -0.25    ,  0.1171875 , -0.07714844,
                0.00521851,  0.125     ,  0.00888733,  0.15527344, -0.02185059,
               -0.15234375, -0.12890625, -0.14765625, -0.13769531, -0.10164062,
```

Gambar 5.42 Praktek 1.7

- berikut merupakan hasil lpengolahan kata car pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [10]: gmodel['car']
Out[10]: array([ 0.13085938,  0.00642285,  0.03344727, -0.05883789,  0.00003906,
                0.14257812,  0.04031641, -0.16894531,  0.20898438,  0.11962891,
                0.18066406, -0.25    , -0.16040391, -0.10742188, -0.01676883,
                0.05200195, -0.00216675,  0.06445312,  0.14453125,  0.04541016,
                0.16113281, -0.01611328, -0.09888379,  0.08447266,  0.16210938,
                0.04467773, -0.15527344,  0.25390625,  0.13984375,  0.00756836,
               -0.25585938, -0.07733398, -0.03295898,  0.16308594, -0.12597656,
               -0.09912109,  0.16503906,  0.06884766, -0.10945312,  0.02032031,
               -0.0534668 , -0.03063965,  0.11083984,  0.24121094, -0.234375  ,
                0.12255536, -0.00294405,  0.1484375 ,  0.13203125,  0.05249023,
               -0.20019531,  0.17695312,  0.12255859,  0.11425781, -0.17675781,
                0.10009766,  0.0039355 ,  0.20757812,  0.20117188,  0.03710938,
                0.11083984, -0.00014453, -0.3125    ,  0.03515625,  0.02032031,
                0.21171875, -0.08866379, -0.02258301, -0.05834961, -0.00787354,
                0.17675781, -0.04296875, -0.17285156,  0.04394531, -0.23046875,
                0.1640625 , -0.12474609, -0.00439173,  0.03196289, -0.10703125,
                0.32617188, -0.04492188, -0.11425781,  0.22851562, -0.01647949,
                0.11035992,  0.13183594,  0.12557656, -0.17480469,  0.02209473,
               -0.0019531 ,  0.00017671,  0.10592766, -0.24609375, -0.1093375 ,
               -0.09375   , -0.01037535, -0.20214844,  0.2314453 , -0.05444336,
               -0.05545992, -0.20898438,  0.20757812,  0.27929688, -0.17480844,
               -0.17578125, -0.02770996, -0.24410156,  0.02392578,  0.03125   ,
               -0.25390625, -0.125     , -0.05493164, -0.17382812, -0.28515625,
               -0.23242188,  0.0234375 , -0.20117188, -0.13476562,  0.35367188,
                0.00769043,  0.20507812,  0.01708984, -0.12988281,  0.04711914,
                0.22070312,  0.02099609, -0.23101562, -0.02091066,  0.17285156,
                0.04272461, -0.10024219, -0.04001906, -0.10992188,  0.10058594,
               -0.09326172,  0.15820312, -0.16503906, -0.00546688,  0.10433594,
               -0.07080078, -0.06684766, -0.09619141, -0.07226562,  0.04882812,
```

Gambar 5.43 Praktek 1.8

- berikut merupakan hasil lpengolahan kata wash pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [11]: gmodel['wash']

Out[11]: array([ 9.46044022e-03,  1.41601562e-01, -5.46875000e-02,  1.14765625e-01,
                -2.38281250e-01,  3.24218750e-01, -6.44726562e-02, -1.39882812e-01,
                 1.07910156e-01,  2.53906250e-01,  1.13525391e-02, -1.66992188e-01,
                -2.75418016e-02,  2.00007812e-01,  4.27246094e-02,  1.05468750e-01,
                -7.62107500e-02,  3.06667500e-01,  2.11914062e-01, -8.88671875e-02,
                 2.67578125e-01,  2.12890625e-01,  1.74609547e-02,  2.02343855e-01,
                 6.29882812e-02,  1.02189375e-01,  1.93359375e-01,  2.17285156e-01,
                -2.67622800e-02, -9.13865938e-02,  2.38281250e-01,  2.23632812e-01,
                 1.24023438e-01,  1.07421875e-01,  1.12304688e-01,  1.13281250e-01,
                 8.78906250e-01,  6.70750938e-02,  1.13128125e-01,  1.66699062e-01,
                -1.05468750e-01,  2.30712891e-02,  2.00195312e-01, -6.84027344e-02,
                -3.45750000e-01, -1.82958701e-01, -3.00955371e-01, -5.85372094e-02,
                 5.07812500e-02,  1.45507812e-01,  2.01250000e-01,  7.61125000e-02,
                 2.84423828e-02, -2.23422188e-01, -5.81854688e-02,  4.51660156e-02,
                -3.56645312e-02,  1.77734375e-01,  1.22070312e-01,  2.71601750e-02,
                -1.18652344e-01,  6.03557750e-02,  2.52685547e-02, -1.17723688e-01,
                 6.21750000e-01,  5.32226562e-02, -3.91578125e-01,  1.74804688e-01,
                 1.77001953e-02, -2.05078125e-02,  2.21679688e-01,  3.18359375e-01,
                 1.08398438e-01, -4.30297852e-03, -2.45317188e-01, -2.80984375e-01,
                 3.58008719e-02,  8.30078125e-02,  1.60945312e-01,  2.79541016e-02,
                 1.04000469e-01, -3.47656250e-01, -5.20019531e-02,  2.24609375e-01,
                 1.69977734e-01,  1.69921875e-01, -1.44484375e-01,  2.66625000e-01,
                 2.17285156e-02,  1.12304688e-02, -1.14257812e-01,  7.22656250e-02,
                 4.52128906e-02,  1.11694336e-02,  5.07554736e-04, -7.91015625e-02,
                -5.98146531e-02, -5.44433594e-02,  3.73046875e-01,  5.62500000e-01,
                -2.20562500e-01, -5.39550781e-02,  1.17705313e-01,  1.84496094e-02,
                -1.55328125e-01, -4.37500000e-01,  2.59765625e-01, -1.49414062e-01,
                 5.66002500e-02,  2.13867188e-01,  2.66862344e-02, -1.74609375e-01,
```

Gambar 5.44 Praktek 1.9

- berikut merupakan hasil lpengolahan kata motor pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [12]: gmodel['motor']

Out[12]: array([ 5.73730469e-02,  1.50390625e-01, -4.61425781e-02, -1.32812500e-01,
                -2.59765625e-01,  1.77734375e-01,  1.60852344e-02,  -0.17500000e-01,
                 2.16375000e-02,  2.57812500e-01,  1.74804688e-01,  2.44140625e-02,
                -2.51953125e-01, -5.70127875e-02,  8.15429688e-02,  1.80767578e-02,
                -3.81300781e-02,  1.58203125e-01,  5.85937500e-02,  1.12304688e-01,
                 1.55250000e-01,  4.24804688e-02, -1.12304688e-01,  2.11914062e-01,
                 1.07968375e-01,  1.69921875e-01,  1.55273438e-01,  4.50984375e-01,
                 2.19921875e-01,  1.07421875e-01,  1.07421875e-01,  1.07421875e-01,
                -3.20120000e-01,  2.18313625e-01,  8.06437500e-02, -7.12890625e-02,
                 1.54290875e-01, -8.88671875e-02,  2.56328125e-01,  5.01523438e-03,
                -4.42773444e-02, -1.86640625e-01,  7.42187500e-02,  5.50197750e-01,
                -1.98593750e-01,  1.80565938e-01,  5.54472656e-02,  2.10937500e-01,
                 3.10062656e-02, -6.50024414e-03,  6.34765625e-02,  4.02812031e-02,
                -2.78328125e-02,  1.07421875e-02,  1.47460938e-01,  2.00767139e-02,
                -1.58390625e-01, -7.12890625e-02,  1.17695312e-01,  9.96093750e-01,
                -3.34672656e-02, -1.88032227e-02, -2.18843750e-01, -0.52148438e-02,
                -6.39648438e-02,  7.51953125e-02, -3.06640625e-01,  2.17773438e-01,
                -2.21679688e-01,  2.33398438e-01,  5.95371094e-02, -3.37890625e-01,
                 1.53328125e-01, -7.12890625e-02,  1.68652344e-02,  7.66601562e-02,
                -8.00781250e-02, -1.14257812e-01, -9.71679688e-02, -2.61718750e-01,
                 3.84765625e-01, -1.87500000e-01, -1.99555562e-01,  1.00593594e-01,
                 1.00390438e-01,  9.57031250e-02, -0.20312500e-02,  1.54296875e-01,
                -2.40234375e-01,  8.34408935e-02,  4.19921875e-02, -1.91958934e-02,
                 9.71679688e-02,  2.52465547e-01,  5.60875000e-02, -5.83378906e-02,
                 8.20312500e-02,  -0.12031250e-01,  -0.18603516,  2.27646484e-02,  5.71289062e-02,
                 1.77734375e-01, -9.57031250e-02,  2.45117188e-01,  0.30447656e-02,
                 1.07910156e-01,  8.03125000e-02,  1.07910156e-01,  1.00390625e-01,
                 4.00390625e-01,  8.03408935e-02, -1.07910156e-01, -1.93115594e-01,
```

Gambar 5.45 Praktek 1.10

- berikut merupakan hasil lpengolahan kata cycle pada data google yang di load. Sehingga memunculkan hasil vektor 300 dimensi untuk kata tersebut.

```
In [13]: gmodel['cycle']

Out[13]: array([ 0.04648016,  0.21670688, -0.02700661,  0.12353516, -0.20703125,
                -0.1328125 ,  0.26367188, -0.12890625, -0.125     ,  0.15332031,
                -0.10202179, -0.15030312, -0.06176758,  0.23972656, -0.15820312,
                 0.02503477, -0.07568359, -0.0625    ,  0.04014258, -0.31054688,
                -0.13379900, -0.13669922, -0.335935  ,  0.078125  ,  0.08447266,
                 0.07225562, -0.06445312,  0.06517578,  0.14041406,  0.13671875,
                 0.10302734,  0.02173852, -0.18603516,  0.02498234, -0.10646515,
                -0.05541992, -0.29492188, -0.40039062,  0.06347656, -0.03447266,
                 0.19778094,  0.01165771, -0.01606772, -0.13671875, -0.16406625 ,
                 0.11425781,  0.20000781, -0.06070102, -0.07275391,  0.15820062,
                 0.10064609, -0.20515625, -0.04052734,  0.03200041,  0.00533110,
                -0.08072002,  0.01564453, -0.39882812,  0.09460938, -0.16843175 ,
                 0.07275391,  0.07275391,  0.07275391,  0.07275391,  0.07275391,
                -0.01171875, -0.07900078,  0.1684375 ,  -0.20214844, -0.01595935,
                 0.09863281, -0.02035574,  0.08789062, -0.07226562, -0.04423828,
                -0.17089844,  0.1406375 ,  0.18546875,  0.06445312,  0.05815494,
                -0.02635719, -0.03686523,  0.125     ,  0.06787200,  0.14225781,
                 0.37109375, -0.15722656,  0.09326172,  0.34390938, -0.00091553,
                -0.03613281,  0.16804531,  0.02656445,  0.07910156, -0.12421875,
                -0.14355469,  0.03173828, -0.07421875,  0.34179688,  0.140625 ,
                 0.0043335 ,  -0.12890625, -0.34960938, -0.02929688, -0.19629906,
                -0.2578125 ,  -0.3671875 ,  0.01463134,  0.20703125,  0.09667969,
                -0.18551562, -0.31054688,  0.02844238,  0.10400391,  0.17773438,
                 0.06689453, -0.1796875 ,  0.02783203,  0.15625 ,  -0.02035317,
                 0.0324707 ,  -0.13476562,  0.15527344,  0.13123812, -0.01095908,
                 0.07958008,  0.01660746,  0.25166016,  0.13379906,  0.02539062,
                 0.10986328, -0.20068469,  0.07275391, -0.35546875, -0.02740582,
```

Gambar 5.46 Praktek 1.11

- berikut merupakan hasil dari similaritas kata kata yang di olah menjadi matrix tadi adapun persentase untuk perbandingan setiap katanya yaitu 9 persen untuk kata wash dan clear 7 persen untuk kata bag dan love 48 persen untuk kata motor dan car 12 persen untuk kata sick dan faith dan terakhir yaitu 6 persen untuk kata cycle dan shine.

```
In [15]: gmodel.similarity('wash', 'clean')
Out[15]: 0.09019176

In [16]: gmodel.similarity('bag', 'love')
Out[16]: 0.07536096

In [17]: gmodel.similarity('motor', 'car')
Out[17]: 0.4810173

In [18]: gmodel.similarity('sick', 'faith')
Out[18]: 0.123073205

In [19]: gmodel.similarity('cycle', 'shine')
Out[19]: 0.061617922
```

Gambar 5.47 Praktek 1.12

2. Jelaskan dengan kata dan ilustrasi fungsi dari extract words dan PermuteSentences

```
In [20]: import re
         def extract_words(sent):
             sent = sent.lower()
             sent = re.sub(r'<[^>]+>', ' ', sent) #hapus tag html
             sent = re.sub(r'(\w)\'(\w)', ' ', sent) #hapus petik satu
             sent = re.sub(r'\W', ' ', sent) #hapus tanda baca
             sent = re.sub(r'\s+', ' ', sent) #hapus spasi yang berurutan
             return sent.split()

In [21]: import random
         class PermuteSentences(object):
             def __init__(self, sents):
                 self.sents = sents

             def __iter__(self):
                 shuffled = list(self.sents)
                 random.shuffle(shuffled)
                 for sent in shuffled:
                     yield sent
```

Gambar 5.48 Praktek 2

3. Jelaskan fungsi dari librari gensim TaggedDocument dan Doc2Vec disertai praktek pemakaiannya.

```
In [22]: from gensim.models.doc2vec import TaggedDocument
         from gensim.models import Doc2Vec

In [ ]: |
```

Gambar 5.49 Praktek 3

4. Jelaskan dengan kata dan praktek cara menambahkan data training dari file yang dimasukkan kepada variabel dalam rangka melatih model doc2vac.

```
In [33]: import os
         sents_sentiments = []

In [34]: for dirname in ["train/pos","train/neg","train/unsup","test/pos","test/neg"]:
             for fname in sorted(os.listdir('aclimdb/'+dirname)):
                 if fname[-4:] == '.txt':
                     with open('aclimdb/'+dirname+'/'+fname,encoding='UTF-8') as f:
                         sent = f.read()
                         words = extract_words(sent)
                         sents_sentiments.append(TaggedDocument(words,[dirname+'/'+fname]))

In [35]: for dirname in ["tst_unsup/pos","tst_unsup/neg"]:
             for fname in sorted(os.listdir(dirname)):
                 if fname[-4:] == '.txt':
                     with open('aclimdb/'+dirname+'/'+fname,encoding='UTF-8') as f:
                         for i, sent in enumerate(f):
                             words = extract_words(sent)
                             sents_sentiments.append(TaggedDocument(words,["%s.%s.%d" % (dirname, fname, i)]))

In [36]: with open('stanfordsentimentTreebank/original_rt_snippets.txt', encoding='UTF-8') as f:
             for i, sent in enumerate(f):
                 words = extract_words(sent)
                 sents_sentiments.append(TaggedDocument(words,["rt-%d" % i]))
```

Gambar 5.50 Praktek 4

5. Jelaskan dengan kata dan praktek kenapa harus dilakukan pengocokan dan pembersihan data.

```
In [77]: import re
         def extract_words(sent):
             sent = sent.lower()
             sent = re.sub(r'<[^>]+>', '', sent) #hapus tag html
             sent = re.sub(r'(\w)\'(\w)', ' ', sent) #hapus pertik satu
             sent = re.sub(r'\W', ' ', sent) #hapus tanda baca
             sent = re.sub(r'\s+', ' ', sent) #hapus spasi yang berurutan
             return sent.split()

         import random
         class PermuteSentences(object):
             def __init__(self, sents):
                 self.sents = sents

             def __iter__(self):
                 shuffled = list(self.sents)
                 random.choice(shuffled)
                 for sent in shuffled:
                     yield sent

         auto-PermuteSentences(unsup_sentences)
         model = Doc2Vec(dm=0, dbow=0, hs=1,size=50)
         model.delete_temporary_training_data(keep_inference=True)
```

Gambar 5.51 Praktek 5

6. Jelaskan dengan kata dan praktek kenapa model harus di save dan kenapa temporari training harus dihapus.

```
In [78]: model.delete_temporary_training_data(keep_inference=True)
         model.save('haci.d2v')
```

Gambar 5.52 Praktek 6

7. Jalankan dengan kta dan praktek maksud dari infer code.

```
In [79]: model.infer_vector(extract_words("I will go home"))

Out[79]: array([-1.7533900e-01,  1.7054952e-01,  8.7283678e-02,  1.7998287e-02,
                 6.2501184e-03, -3.2540500e-02,  1.6391091e-01, -1.3780003e-02,
                 5.1754227e-02,  4.6472001e-02,  1.6690210e-02,  2.0200551e-02,
                -2.5965100e-02,  7.7927085e-02, -1.5400573e-02,  1.3232067e-01,
                -4.3078404e-02,  8.8150723e-02, -5.1151483e-02, -7.2081677e-02,
                 3.1809944e-02,  2.4905023e-02,  2.1035837e-01,  7.9181477e-02,
                 1.1294000e-01, -1.4000000e-01,  1.1000000e-01,  1.1000000e-01,
                 1.6294051e-02, -6.1351705e-02,  1.3731884e-01, -7.8027303e-02,
                -1.4081554e-01,  4.0382754e-02, -1.2432594e-01, -3.1883365e-02,
                 6.4639558e-02,  7.1126500e-02, -4.4144000e-02, -5.9013160e-02,
                 1.0170090e-04,  1.0074181e-02,  6.5020474e-02,  3.7513527e-03,
                 5.1376358e-02, -1.3092435e-02, -5.1841831e-02,  4.9701292e-02,
                -1.1787581e-01,  9.5305327e-02], dtype=float32)
```

Gambar 5.53 Praktek 7

8. Jelaskan dengan praktek dan kata maksud dari cosine similarity.

```
In [83]: from sklearn.metrics.pairwise import cosine_similarity
         cosine_similarity(
             [model.infer_vector(extract_words("Now going to school, after wash hand")),
             [model.infer_vector(extract_words("Services sucks2.."))])

Out[83]: array([[0.0474403]], dtype=float32)

In [84]: from sklearn.metrics.pairwise import cosine_similarity
         cosine_similarity(
             [model.infer_vector(extract_words("Dia pergi ke sekolah tadi siang")),
             [model.infer_vector(extract_words("Services sucks2.."))])

Out[84]: array([[0.2005401]], dtype=float32)
```

Gambar 5.54 Praktek 8

9. Jelaskan dengan praktek score dari cross validation masing-masing metode.

```
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()

scores = cross_val_score(clf, sentvecs, sentiments, cv=5)
print((np.mean(scores), np.std(scores)))

```

```
scores = cross_val_score(clfrf, sentvecs, sentiments, cv=5)
print((np.mean(scores), np.std(scores)))

# bag-of-words comparison
from sklearn.pipeline import make_pipeline
from sklearn.feature_extraction.text import CountVectorizer,
    TfidfTransformer
pipeline = make_pipeline(CountVectorizer(), TfidfTransformer(),
    RandomForestClassifier())
scores = cross_val_score(pipeline, sentences, sentiments, cv=5)
print((np.mean(scores), np.std(scores)))
```

5.2.3 Penangan Error

1. SS Error

Gambar 5.55 Name Error

2. Jenis error
   - Name Error
3. Cara Penanganan
   Dengan Mengecek Kembali Baris Code Yang Dibuat

5.2.4 Bukti Tidak Melakukan Plagiat

Gambar 5.56 Bukti Tidak Melakukan Plagiat

BAB 6

CHAPTER 6

6.1 1174069 - Fanny Shafira Damayanti

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.

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

Gambar 6.1 Contoh MFCC

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

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 :

Gambar 6.3 Contoh Pembobotan Neural Network

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

Gambar 6.4 Cara Membaca Hasil Plot MFCC

   Dari gambar tersebut dapat 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

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
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

1. Penjelasan isi data GTZAN Genre Collection dan data dari Freesound.

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

def display_mfcc(song):
    y, _ = librosa.load(song)
    mfcc = librosa.feature.mfcc(y)

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

2. Penjelasan perbaris kode program dari display MFCC.

```
display_mfcc('test1.wav')
display_mfcc('test2.wav')
display_mfcc('dataset/genres/disco/disco.00069.au')
display_mfcc('dataset/genres/blues/blues.00069.au')
display_mfcc('dataset/genres/classical/classical.00069.au')
display_mfcc('dataset/genres/country/country.00069.au')
display_mfcc('dataset/genres/hiphop/hiphop.00069.au')
display_mfcc('dataset/genres/jazz/jazz.00069.au')
display_mfcc('dataset/genres/pop/pop.00069.au')
display_mfcc('dataset/genres/reggae/reggae.00069.au')
display_mfcc('dataset/genres/rock/rock.00069.au')
```

3. Penjelasan perbaris code dari Extract Feature Song.

```
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]
```

4. Penjelasan perbaris code dari Generate Features and Labels.

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

```

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

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

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

```
features, labels = generate_features_and_labels()
print(np.shape(features))
print(np.shape(labels))
```

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

```
training_split = 0.8
alldata = np.column_stack((features, labels))
np.random.shuffle(alldata)
splitidx = int(len(alldata) * training_split)
train, test = alldata[:splitidx,:], alldata[splitidx:,:]
print(np.shape(train))
print(np.shape(test))
train_input = train[:,:-10]
train_labels = train[:,-10:]
test_input = test[:,:-10]
test_labels = test[:,-10:]
print(np.shape(train_input))
print(np.shape(train_labels))
```

7. Parameter dari fungsi Sequensial().

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

8. Parameter dari fungsi Compile().

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

9. Parameter dari fungsi Fit().

```
model.fit(train_input, train_labels, epochs=10, batch_size=32,
          validation_split=0.2)
```

10. Parameter dari fungsi Evaluate().

```
loss, acc = model.evaluate(test_input, test_labels, batch_size
    =32)
print("Done!")
print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

11. Parameter dari fungsi Predict().

```
model.predict(test_input[:1])
```

6.1.3 Penanganan Error

1. SS Error

Gambar 6.9 No Module Name error

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

6.1.4 Bukti Tidak Plagiat

Gambar 6.10 Tidak Melakukan Plagiat Pada Ch 6

6.1.5 Link Youtube

https://youtu.be/eeLATXvf8bE

6.2 1174086 - Tia Nur Candida

6.2.1 Teori

1. Kenapa file suara harus di lakukan MFCC. dilengkapi dengan ilustrasi atau gambar.
   Mampu untuk menangkap karakteristik suara yang sangat penting bagi pengenalan suara, atau dengan kata lain dapat menangkap informasi-informasi penting yang terkandung dalam signal suara. Menghasilkan data seminimal mungkin, tanpa menghilangkan informasi-informasi penting yang dikandungnya. Mereplikasi organ pendengaran manusia dalam melakukan persepsi terhadap signal suara.

Gambar 6.11 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 6.12 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 :

Gambar 6.13 Contoh Pembobotan Neural Network

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

Gambar 6.14 Cara Membaca Hasil Plot MFCC

Dari gambar tersebut dapat 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

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

Gambar 6.15 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 6.16 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 6.17 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 6.18 Sequential

6.2.2 Praktek

1. Penjelasan isi data GTZAN Genre Collection dan data dari Freesound.

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

def display_mfcc(song):
    y, _ = librosa.load(song)
    mfcc = librosa.feature.mfcc(y)

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

2. Penjelasan perbaris kode program dari display MFCC.

```
display_mfcc('test1.wav')
display_mfcc('test2.wav')
display_mfcc('dataset/genres/disco/disco.00069.au')
display_mfcc('dataset/genres/blues/blues.00069.au')
display_mfcc('dataset/genres/classical/classical.00069.au')
display_mfcc('dataset/genres/country/country.00069.au')
display_mfcc('dataset/genres/hiphop/hiphop.00069.au')
display_mfcc('dataset/genres/jazz/jazz.00069.au')
display_mfcc('dataset/genres/pop/pop.00069.au')
display_mfcc('dataset/genres/reggae/reggae.00069.au')
display_mfcc('dataset/genres/rock/rock.00069.au')
```

3. Penjelasan perbaris code dari Extract Feature Song.

```
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]
```

4. Penjelasan perbaris code dari Generate Features and Labels.

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

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

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

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

```
features, labels = generate_features_and_labels()
print(np.shape(features))
print(np.shape(labels))
```

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

```
training_split = 0.8
alldata = np.column_stack((features, labels))
np.random.shuffle(alldata)
splitidx = int(len(alldata) * training_split)
train, test = alldata[:splitidx,:], alldata[splitidx:,:]
print(np.shape(train))
print(np.shape(test))
train_input = train[:,:-10]
train_labels = train[:,-10:]
test_input = test[:,:-10]
```

```
test_labels = test[:,-10:]
print(np.shape(train_input))
print(np.shape(train_labels))
```

7. Parameter dari fungsi Sequensial().

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

8. Parameter dari fungsi Compile().

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

9. Parameter dari fungsi Fit().

```
model.fit(train_input, train_labels, epochs=10, batch_size=32,
          validation_split=0.2)
```

10. Parameter dari fungsi Evaluate().

```
loss, acc = model.evaluate(test_input, test_labels, batch_size
    =32)
print("Done!")
print("Loss: %.4f, accuracy: %.4f" % (loss, acc))
```

11. Parameter dari fungsi Predict().

```
model.predict(test_input[:1])
```

6.2.3 Penanganan Error

1. SS Error

Gambar 6.19 No Module Name error

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

6.2.4 Bukti Tidak Plagiat

Gambar 6.20 Tidak Melakukan Plagiat Pada Ch 6

BAB 7

CHAPTER 7

7.1 1174006 - Kadek Diva Krishna Murti

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

```
@inproceedings{awangga2017colenak,
  title={Colenak: GPS tracking model for post-stroke rehabilitation program using AES-CBC URL encryption and QR-Code},
  author={Awangga, Rolly Maulana and Fathonah, Nuraini Siti and Hasanudin, Trisna Irmayadi},
  booktitle={Information Technology, Information Systems and Electrical Engineering (ICITISEE), 2017 2nd International conferences on},
  pages={255--260},
  year={2017},
  organization={IEEE}
}
```


By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

Gambar 7.1 Kecerdasan Buatan.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
2. Lorem ipsum dolor sit amet, consectetur adipiscing elit.
3. Lorem ipsum dolor sit amet, consectetur adipiscing elit.

7.1.1 Teori

7.1.2 Praktek

7.1.3 Penanganan Error

7.1.4 Bukti Tidak Plagiat

Gambar 7.2 Kecerdasan Buatan.

7.2 1174066 - D.Irga B. Naufal Fakhri

7.2.1 Teori

7.2.1.1 Kenapa file teks harus di lakukan tokenizer

Karena MTokenizer adalah proses membagi teks yang berupa kalimat, paragraf atau dokumen menjadi kata-kata atau bagian-bagian tertentu dalam kalimat tersebut. Contohnya kalimat ”Tugas chapter 7”, kalimat itu menjadi beberapa bagian yaitu ”Tugas”, ”chapter”, ”7”. Yang menjadi acuan yakni tanda baca dan spasi.

Gambar 7.3 Teori 1

7.2.1.2 konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1.

Konsep sederhana dari K Fold Cross Validation ialah Pada code ini:

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

terdapat kfold yang bertujuan untuk melakukan split data menjadi 5 bagian dari dataset komentar Youtube tersebut. Sehingga dari setiap data yang sudah dibagi tersebut akan menghasilkan presentase dari setiap bagiannya, untuk menghasilkan hasil akhir dengan presentase yang cukup baik.

Gambar 7.4 Teori 2

7.2.1.3 Apa maksudnya kode program for train, test in splits

For train berfungsi untuk membagi data tersebut menjadi data training. Sedangkan test in splits berfungsi untuk menguji apakah dataset tersebut sudah dibagi menjadi beberapa bagian atau masih menumpuk.

Gambar 7.5 Teori 3

7.2.1.4 Apa maksudnya kode program train content = d[â^CONTENTâ^].iloc[train idx] dan test content = d[â^CONTENTâ^].iloc[test idx]

Fungsi dalam kode tersebut berfungsi untuk mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Contoh sederhananya ketika data telah diubah menjadi data train dan data test maka kita dapat memilihnya untuk ditampilkan pada kolom yang di inginkan.

```
import pandas as pd

mydict = [{'a': 1, 'b': 2, 'c': 3, 'd': 4},
          {'a': 100, 'b': 200, 'c': 300, 'd': 400},
          {'a': 1000, 'b': 2000, 'c': 3000, 'd': 4000 }]
df = pd.DataFrame(mydict)
df
```

```
Out[2]:
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Gambar 7.6 Teori 4

7.2.1.5 Apa maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content)

Fungsi tokenizer berfungsi untuk melakukan vektorisasi data kedalam bentuk token sebanyak 2000 kata. Dan selanjutnya akan melakukan fit tokenizer hanya untuk data training saja tidak dengan data testingnya.

Gambar 7.7 Teori 5

7.2.1.6 Apa maksud dari fungsi d train inputs = tokenizer.texts to matrix(train content, mode=â^tfidf â^) dan d test inputs = tokenizer.texts to matrix(test content, mode=â^tfidf â^)

Maksud dari baris diatas ialah untuk memasukkan text ke sebuah matrix dengan mode tfidf dan menginputkan data testing untuk di terjemahkan ke sebuah matriks.

Gambar 7.8 Teori 6

7.2.1.7 Apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train inputs)) dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs))

Fungsi np.amax adalah nilai Maksimal. Jika sumbu tidak ada, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi a.ndim - 1.

Gambar 7.9 Teori 7

7.2.1.8 Apa maksud fungsi dari d train outputs = np utils.to categorical(d[â^CLASSâ^].ilo idx]) dan d test outputs = np utils.to categorical(d[â^CLASSâ^].iloc[test idx]) dalam kode program

Fungsi dari baris kode tersebut ialah membuat train outputs dengan kategori dari class lalu dengan ketentuan iloc train idx. Kemudian membuat keluaran sebagai output.

Gambar 7.10 Teori 8

7.2.1.9 Apa maksud dari fungsi di listing 7.2.

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

Fungsi dari baris kode tersebut ialah model perlu mengetahui bentuk input apa yang harus diharapkan. Untuk alasan ini, lapisan pertama dalam model Sequential (dan hanya yang pertama, karena lapisan berikut dapat melakukan inferensi bentuk otomatis) perlu menerima informasi tentang bentuk inputnya.

Gambar 7.11 Teori 9

7.2.1.10 Apa maksud dari fungsi di listing 7.3 dengan parameter tersebut

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

Fungsi dari baris kode tersebut ialah bisa meneruskan nama fungsi loss yang ada, atau melewati fungsi simbolis TensorFlow yang mengembalikan skalar untuk setiap titik data dan mengambil dua argumen y_true: True label. dan y_pred: Prediksi. Tujuan yang dioptimalkan sebenarnya adalah rata-rata dari array output di semua titik data.

Gambar 7.12 Teori 10

7.2.1.11 Apa itu Deep Learning

Deep Learning adalah salah satu cabang dari ilmu machine learning yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam machine learning dapat digunakan baik untuk supervised learning dan unsupervised learning.

7.2.1.12 Apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning

Deep Neural Network adalah salah satu algoritma berbasis jaringan saraf tiruan yang memiliki dari 1 lapisan saraf tersembunyi yang dapat digunakan untuk pengambilan keputun. Perbedaannya dengan deep learning, yakni: Deep Neural Network dapat menentukan dan mencerna karakteristik tertentu di suatu rangkaian data, kapabilitas lebih kompleks untuk mempelajari, mencerna, dan mengklasifikasikan data, serta dibagi ke dalam berbagai lapisan dengan fungsi yang berbeda-beda.

7.2.1.13 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.

Konvolusi terdapat pada operasi pengolahan citra yang mengalikan sebuah citra dengan sebuah mask atau kernel, Stride adalah parameter yang berfungsi untuk menentukan pergeseran pada filter data pixel yang terjadi. untuk contoh penggunaannya:

Gambar 7.13 Teori 11

7.2.2 Praktek

7.2.2.1 Nomor 1

```
import csv #Import library csv
from PIL import Image as pil_image #Import library Image yaitu fungsi PIL (Python Imaging Library) yang berguna untuk mengolah data berupa gambar
import keras.preprocessing.image #Import library keras yang menggunakan method preprocessing yang digunakan untuk membuat neural network
```

Hasil:

Gambar 7.14 Hasil No 1

7.2.2.2 Nomor 2

```
imgs = [] #Membuat variabel imgs
classes = [] #Membuat variabel classes dengan variabel kosong
with open('N:/HASYv2/hasy-data-labels.csv') as csvfile: #Membuka file hasy-data-labels.csv
    csvreader = csv.reader(csvfile) #Membuat variabel csvreader yang berisi method csv.reader untuk membaca csvfile
    i = 0 # membuat variabel i dengan isi 0
    for row in csvreader: # Membuat looping pada variabel csvreader
        if i > 0: #Ketentuannya jika i lebih kecil daripada 0
            img = keras.preprocessing.image.img_to_array(pil_image.open("N:/HASYv2/" + row[0])) #Dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            # 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 #Membagi data yang ada pada fungsi img sebanyak 255.0
            imgs.append((row[0], row[2], img)) #Menambah nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            classes.append(row[2]) #Menambahkan nilai pada row ke 2 pada variabel classes
        i += 1 #Menambah nilai satu pada variabel i
```

Hasil:

Gambar 7.15 Hasil No 2

7.2.2.3 Nomor 3

```
import random #Mengimport library random
random.shuffle(imgs) #Melakukan randomize pada fungsi imgs
split_idx = int(0.8*len(imgs)) #Membuat variabel split_idx dengan
    nilai integer 80 persen dikali dari pengembalian jumlah dari
    variabel imgs
train = imgs[:split_idx] #Membuat variabel train dengan isi sebelum
    split idx
test = imgs[split_idx:] #Membuat variabel test dengan isi setelah
    split idx
```

Hasil:

Gambar 7.16 Hasil No 1

7.2.2.4 Nomor 4

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

Hasil:

Gambar 7.17 Hasil No 4

7.2.2.5 Nomor 5

```
from sklearn.preprocessing import LabelEncoder #Mengimport library LabelEncode dari sklearn
from sklearn.preprocessing import OneHotEncoder #Mengimport library OneHotEncoder dari sklearn
```

Hasil:

Gambar 7.18 Hasil No 5

7.2.2.6 Nomor 6

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

Hasil:

Gambar 7.19 Hasil No 6

7.2.2.7 Nomor 7

```
onehot_encoder = OneHotEncoder(sparse=False)#Membuat variabel onehot_encoder dengan isi OneHotEncoder
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1) #Mengisi variabel integer_encoded dengan isi integer_encoded yang telah di convert pada fungsi sebelumnya
onehot_encoder.fit(integer_encoded) #Mengkonvert variabel integer_encoded kedalam onehot_encoder
```

Hasil:

Gambar 7.20 Hasil No 7

7.2.2.8 Nomor 8

```
train_output_int = label_encoder.transform(train_output) #Mengkonvert data train output  mengguanakn variabel label_encoder kedalam variabel train_output_int
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1)) #Mengkonvert variabel train_output_int kedalam fungsi onehot_encoder
```

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

Hasil:

Gambar 7.21 Hasil No 8

7.2.2.9 Nomor 9

```
from keras.models import Sequential #Mengimport library Sequential dari Keras
from keras.layers import Dense, Dropout, Flatten #Mengimport library Dense, Dropout, Flatten dari Keras
from keras.layers import Conv2D, MaxPooling2D #Mengimport library Conv2D, MaxPooling2D dari Keras
```

Hasil:

Gambar 7.22 Hasil No 9

7.2.2.10 Nomor 10

```
model = Sequential() #Membuat variabel model dengan isian library Sequential
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0]))) #Variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(MaxPooling2D(pool_size=(2, 2))) #Variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixel
```

```
model.add(Conv2D(32, (3, 3), activation='relu')) #Variabel model di tambahkan dengan library Conv2D 32bit dengan kernel 3 x 3
model.add(MaxPooling2D(pool_size=(2, 2))) #Variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(Flatten()) #Variabel model di tambahkan library Flatten
model.add(Dense(1024, activation='tanh')) #Variabel model di tambahkan library Dense dengan fungsi tanh
model.add(Dropout(0.5)) #Variabel model di tambahkan library dropout untuk memangkas data tree sebesar 50 persen
model.add(Dense(num_classes, activation='softmax')) #Variabel model di tambahkan library Dense dengan data dari num_classes dan fungsi softmax
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy']) #Mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
print(model.summary()) #Mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
```

Hasil:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
 max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)       0
_________________________________________________________________
 conv2d_2 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
 max_pooling2d_2 (MaxPooling2 (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 7.23 Hasil No 10

7.2.2.11 Nomor 11

```
import keras.callbacks #Mengimport library keras dengan fungsi callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='N:/KB/hasyv2/logs/mnist-style') #Membuat variabel tensorboard dengan isi lib keras
```

Hasil:

```
[11] import keras.callbacks #Mengimport library keras dengan fungsi callbacks
     tensorboard = keras.callbacks.TensorBoard(log_dir='/content/logs/mnist-style')
```

Gambar 7.24 Hasil No 11

7.2.2.12 Nomor 12

```
model.fit(train_input, train_output, #Fungsi model ditambahkan fungsi fit untuk mengetahui perhitungan dari train_input train_output
          batch_size=32, #Dengan batch size 32 bit
```

```
            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])
```

Hasil:

Gambar 7.25 Hasil No 12

7.2.2.13 Nomor 13

```
import time #Mengimport library time
results = [] #Membuat variabel result dengan array kosong
for conv2d_count in [1, 2]: #Melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
    for dense_size in [128, 256, 512, 1024, 2048]: #Menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
        for dropout in [0.0, 0.25, 0.50, 0.75]: #Membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
            model = Sequential() #Membuat variabel model Sequential
            for i in range(conv2d_count): #Membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
                if i == 0: #Syarat jika i samadengan bobotnya 0
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0]))) #Menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                else: #Jika tidak
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu')) #Menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                model.add(MaxPooling2D(pool_size=(2, 2))) #Menambahkan method add pada variabel model dengan isian method Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
            model.add(Flatten()) #Merubah feature gambar menjadi 1 dimensi vektor
```

```
        model.add(Dense(dense_size, activation='tanh')) # Menambahkan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
        if dropout > 0.0: #Membuat ketentuan jika pemangkasan lebih besar dari 0 persen
            model.add(Dropout(dropout)) #Menambahkan method dropout pada model dengan nilai dari dropout
        model.add(Dense(num_classes, activation='softmax')) # Menambahkan method dense dengan fungsi num classs dan rumus softmax
        model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy']) #Mengcompile variabel model dengan hasi loss optimasi dan akurasi matrix
        log_dir = 'N:/KB/hasyv2/logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout) #Melakukan log
        tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir) # membuat variabel tensorboard dengan isian dari library keras dan nilai dari log_dir

        start = time.time() #Membuat variabel start dengan isian dari library time menggunakan method time
        model.fit(train_input, train_output, batch_size=32, epochs=10,
                  verbose=0, validation_split=0.2, callbacks=[tensorboard]) #Menambahkan method fit pada model dengan data dari train input train output nilai batch nilai epoch verbose nilai 20 persen validation split dan callback dengan nilai tensorboard.
        score = model.evaluate(test_input, test_output, verbose=2) #Membuat variabel score dengan nilai evaluasi dari model menggunakan data tes input dan tes output
        end = time.time() #Membuat variabel end
        elapsed = end - start  #Membuat variabel elapsed
        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)) #Mencetak hasil perhitungan
        results.append((conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
```

Hasil:

Gambar 7.26 Hasil No 13

7.2.2.14 Nomor 14

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

Hasil:

```
Model: "sequential_42"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_63 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_63 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_64 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_64 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
flatten_42 (Flatten)         (None, 1152)              0
_________________________________________________________________
dense_83 (Dense)             (None, 128)               147584
_________________________________________________________________
dropout_32 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_84 (Dense)             (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.27 Hasil No 14

7.2.2.15 Nomor 15

```
model.fit(np.concatenate((train_input, test_input)), #Melakukan training yang isi datanya dari join numpy menggunakan data train_input test_input
          np.concatenate((train_output, test_output)), #Kelanjutan data yang di gunakan pada join train_output test_output
```

```
3            batch_size=32, epochs=10, verbose=2) #Menggunakan ukuran 32
     bit dan epoch 10
```

Hasil:

Gambar 7.28 Hasil No 15

7.2.2.16 Nomor 16

```
1 model.save("mathsymbols.model") #Menyimpan model atau mengeksport
     model yang telah di jalan tadi
```

Hasil:

Gambar 7.29 Hasil No 16

7.2.2.17 Nomor 17

```
1 np.save('classes.npy', label_encoder.classes_) #Menyimpan label
     encoder dengan nama classes.npy
```

Hasil:

Gambar 7.30 Hasil No 17

7.2.2.18 Nomor 18

```
1 import keras.models #Mengimpport library keras model
2 model2 = keras.models.load_model("mathsymbols.model") #Membuat
     variabel model2 untuk meload model yang telah di simpan tadi
3 print(model2.summary()) #Mencetak hasil model2
```

Hasil:

```
Model: "sequential_42"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_63 (Conv2D)          (None, 30, 30, 32)        896
_________________________________________________________________
 max_pooling2d_63 (MaxPooling (None, 15, 15, 32)       0
_________________________________________________________________
 conv2d_64 (Conv2D)          (None, 13, 13, 32)        9248
_________________________________________________________________
 max_pooling2d_64 (MaxPooling (None, 6, 6, 32)         0
_________________________________________________________________
 flatten_42 (Flatten)        (None, 1152)              0
_________________________________________________________________
 dense_83 (Dense)            (None, 128)               147584
_________________________________________________________________
 dropout_32 (Dropout)        (None, 128)               0
_________________________________________________________________
 dense_84 (Dense)            (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.31 Hasil No 18

7.2.2.19 Nomor 19

```
label_encoder2 = LabelEncoder() # membuat variabel label encoder ke 2 dengan isian fungsi label encoder.
label_encoder2.classes_ = np.load('classes.npy') #Menambahkan method classess dengan data classess yang di eksport tadi
def predict(img_path): #Membuat fumgsi predict dengan path img
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path)) #Membuat variabel newimg dengam membuay immage menjadi array dan membuka data berdasarkan img path
    newimg /= 255.0 #Membagi data yang terdapat pada variabel newimg sebanyak 255

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

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

Hasil:

```
[19] label_encoder2 = LabelEncoder() # membuat variabel label encoder ke 2 dengan isian
     label_encoder2.classes_ = np.load('/content/classes.npy') #Menambahkan method clas
     def predict(img_path): #Membuat fumgsi predict dengan path img
         newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path)) #Me
         newimg /= 255.0 #Membagi data yang terdapat pada variabel newimg sebanyak 255

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

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

Gambar 7.32 Hasil No 19

7.2.2.20 Nomor 20

```
predict("HASYv2/hasy-data/v2-00010.png") #Mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00010.png
predict("HASYv2/hasy-data/v2-00500.png") #Mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00500.png
predict("HASYv2/hasy-data/v2-00700.png") #Mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00700.png
```

Hasil:

```
Prediction: A, confidence: 0.74
Prediction: \pi, confidence: 0.48
Prediction: \alpha, confidence: 0.90
```

Gambar 7.33 Hasil No 20

7.2.3 Penanganan Error

7.2.3.1 Error

- ProfilerNotRunningError

```
ProfilerNotRunningError: Cannot stop profiling. No profiler is running.
```

Gambar 7.34 ProfilerNotRunningError

7.2.3.2 Solusi Error

- ProfilerNotRunningError
  tambahkan kode profile=10000000 pada parameter log_dir

7.2.4 Bukti Tidak Plagiat

Gambar 7.35 Bukti tidak plagiat

7.2.5 Link Youtube

https://youtu.be/Vq_LZ89hTpg

7.3 1174080 - Handi Hermawan

7.3.1 Soal Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
   Tokenizer untuk memisahkan input text menjadi potongan yang memiliki arti disebut tokenization. Hasil dari proses tokenization disebut token. Token dapat berupa kata, kalimat, paragraph dan lainnya. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.36 Teori 1

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

```

kfold = StratifiedKFold(n_splits=5)
```

   Konsep sederhana dari K Fold Cross Validation ialah Pada code tesebut terdapat kfold yang bertujuan untuk melakukan split data menjadi 5 bagian dari dataset komentar Youtube tersebut. Sehingga dari setiap data yang sudah dibagi tersebut akan menghasilkan presentase dari setiap bagiannya, untuk menghasilkan hasil akhir dengan presentase yang cukup baik. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.37 Teori 2

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

   Untuk penjelasan nya yaitu For train berfungsi untuk membagi data tersebut menjadi data training. Sedangkan test in splits berfungsi untuk menguji apakah dataset tersebut sudah dibagi menjadi beberapa bagian atau masih menumpuk. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.38 Teori 3

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.

   Fungsi dalam kode tersebut berfungsi untuk mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Contoh sederhananya ketika data telah diubah menjadi data train dan data test maka kita dapat memilihnya untuk ditampilkan pada kolom yang di inginkan. Namun, untuk ilustrasi lihat gambar berikut:

```
import pandas as pd

mydict = [{'a': 1, 'b': 2, 'c': 3, 'd': 4},
          {'a': 100, 'b': 200, 'c': 300, 'd': 400},
          {'a': 1000, 'b': 2000, 'c': 3000, 'd': 4000 }]
df = pd.DataFrame(mydict)
df
```

```
Out[5]:
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Gambar 7.39 Teori 4

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 ini berfungsi untuk melakukan vektorisasi data kedalam bentuk token sebanyak 2000 kata. Dan selanjutnya akan melakukan fit tokenizer hanya untuk data training saja tidak dengan data testingnya. Untuk ilustrasi lihat gambar berikut:

Gambar 7.40 Teori 5

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.
   Maksud dari baris diatas ialah untuk memasukkan text ke sebuah matrix dengan mode tfidf dan menginputkan data testing untuk di terjemahkan ke sebuah matriks. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.41 Teori 6

7. Jelaskan apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train inputs)) dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs)), dilengkapi dengan ilustrasi atau gambar.
   Fungsi np.amax adalah nilai Maksimal. Jika sumbu tidak ada, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi a.ndim - 1. Untuk ilustrasi, lihat gambar berikut:

```
>>> a = np.arange(4).reshape((2,2))
>>> a
array([[0, 1],
       [2, 3]])
>>> np.amax(a)           # Maximum of the flattened array
3
>>> np.amax(a, axis=0)   # Maxima along the first axis
array([2, 3])
>>> np.amax(a, axis=1)   # Maxima along the second axis
array([1, 3])
>>> np.amax(a, where=[False, True], initial=-1, axis=0)
array([-1,  3])
>>> b = np.arange(5, dtype=float)
>>> b[2] = np.NaN
>>> np.amax(b)
nan
>>> np.amax(b, where=~np.isnan(b), initial=-1)
4.0
>>> np.nanmax(b)
4.0
```

Gambar 7.42 Teori 7

8. Jelaskan apa maksud fungsi dari d train outputs = np utils.to categorical(d[CLASS].iloc[tra idx]) dan d test outputs = np utils.to categorical(d[CLASS].iloc[test idx]) dalam kode program, dilengkapi dengan ilustrasi atau gambar.

Fungsi dari baris kode tersebut ialah membuat train outputs dengan kategori dari class lalu dengan ketentuan iloc train idx. Kemudian membuat keluaran sebagai output. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.43 Teori 8

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

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

Fungsi dari baris kode tersebut ialah model perlu mengetahui bentuk input apa yang harus diharapkan. Untuk alasan ini, lapisan pertama dalam model Sequential (dan hanya yang pertama, karena lapisan berikut dapat melakukan inferensi bentuk otomatis) perlu menerima informasi tentang bentuk inputnya.

Gambar 7.44 Teori 9

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

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

Fungsi dari baris kode tersebut ialah bisa meneruskan nama fungsi loss yang ada, atau melewati fungsi simbolis TensorFlow yang mengembalikan skalar untuk setiap titik data dan mengambil dua argumen y_true: True label. dan y_pred:

Prediksi. Tujuan yang dioptimalkan sebenarnya adalah rata-rata dari array output di semua titik data.

Gambar 7.45 Teori 10

11. Jelaskan apa itu Deep Learning.

    Deep Learning adalah salah satu cabang dari ilmu machine learning yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam machine learning dapat digunakan baik untuk supervised learning dan unsupervised learning.

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning.

    DNN adalah salah satu algoritma berbasis jaringan saraf tiruan yang memiliki dari 1 lapisan saraf tersembunyi yang dapat digunakan untuk pengambilan keputun. Perbedaannya dengan deep learning, yakni: DNN dapat menentukan dan mencerna karakteristik tertentu di suatu rangkaian data, kapabilitas lebih kompleks untuk mempelajari, mencerna, dan mengklasifikasikan data, serta dibagi ke dalam berbagai lapisan dengan fungsi yang berbeda-beda.

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.

    1174080 mod3+1 x 1174080 mod3+1 = 2 x 2, adapun ilustrasi gambar nya sebagai berikut :

Gambar 7.46 Teori 9

7.3.2 Praktek Program

1. Soal 1

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

Kode di atas menjelaskan tentang library yang akan di pakai yaitu import file csv, lalu load module pil image dan juga import library keras, hasilnya adalah sebagai berikut:

Gambar 7.47 Hasil Soal 1.

2. Soal 2

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

```
            #membaji data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # menambah nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # menambahkan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

Kode di atas akan menampilkan hasil dari proses load dataset dari HASYv2 sebagai file csv, membuat variabel i dengan parameter 0, lalu perintah perulangan if dengan i¿0, variabel img dengan nilai 255.0, imgs.append yaitu proses melampirkan atau menggabungkan data dengan file. Berikut adalah hasil setelah saya lakukan running dan pembacaan file audio :

3. Soal 3

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

Perintah import random yaitu sebagai library untuk menghasilkan nilai acak, disini kita membuat data menjadi 2 bagian yaitu 80% sebagai training dan 20% sebagai data testing. Hasilnya adalah sebagai berikut :

Gambar 7.48 Hasil Soal 3.

4. Soal 4

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

```
# membuat test input input dengan np method asarray yang mana membuat array dengan isi row 2 dari data test
test_input = np.asarray(list(map(lambda row: row[2], test)))
# membuat variabel train_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data train
train_output = np.asarray(list(map(lambda row: row[1], train)))
# membuat variabel test_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data test
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Kode di atas dapat digunakan untuk melakukan fungsi yang sebelumnya telah kita lakukan yaitu dengan membuat variabel baru untuk menampung data train input dan test input lalu keluaran nya sebagai output, . Hasilnya adalah sebagai berikut :

Gambar 7.49 Hasil Soal 4.

5. Soal 5

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

Kode diatas untuk melakukan import library yang berfungsi sebagai label encoder dan one hot encoder. Hasilnya adalah sebagai berikut :

Gambar 7.50 Hasil Soal 5.

6. Soal 6

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

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

Kode diatas berfungsi untuk melakukan convert class names ke one-hot encoding. Hasilnya adalah sebagai berikut :

```
In [47]: label_encoder = LabelEncoder()
   ...: # membuat variabel integer_encoded yang
berfungsi untuk mengkonvert variabel classes kedalam
bentuk integer
   ...: integer_encoded =
label_encoder.fit_transform(classes)
```

Gambar 7.51 Hasil Soal 6.

7. Soal 7

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

Kode di atas befungsi untuk melakukan convert integers ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

```
In [8]: onehot_encoder = OneHotEncoder(sparse=False)
   ...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
   ...: onehot_encoder.fit(integer_encoded)
Out[8]:
OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
              handle_unknown='error', sparse=False)
```

Gambar 7.52 Hasil Soal 7.

8. Soal 8

```
# In[8]:convert train and test output to one-hot
# mengkonvert data train output  mengguanakn variabel
    label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# mengkonvert variabel train_output_int kedalam fungsi
    onehot_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
# mengkonvert data test_output mengguanakn variabel label_encoder
     kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# mengkonvert variabel test_output_int kedalam fungsi
    onehot_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
# membuat variabel num_classes dengan isi variabel label_encoder
    dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

Kode di atas befungsi untuk melakukan convert train dan test output ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

Gambar 7.53 Hasil Soal 8.

9. Soal 9

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

Kode di atas befungsi untuk melakukan import sequential dari library keras. Hasilnya adalah sebagai berikut :

Gambar 7.54 Hasil Soal 9.

10. Soal 10

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

```
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

Kode di atas befungsi untuk melihat detail desain pada jaringan model yang telah di defenisikan. Hasilnya adalah sebagai berikut :

Gambar 7.55 Hasil Soal 10.

11. Soal 11

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

Kode di atas befungsi untuk melakukan load callbacks pada library keras. Hasilnya adalah sebagai berikut :

Gambar 7.56 Hasil Soal 11.

12. Soal 12

```
# In[12]: 5menit kali 10 epoch = 50 menit
# fungsi model titambahkan metod fit untuk mengetahui perhitungan dari train_input train_output
model.fit(train_input, train_output,
```

```
# dengan batch size 32 bit
            batch_size=32,
            epochs=10,
            verbose=2,
            validation_split=0.2,
            callbacks=[tensorboard])

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

Kode di atas befungsi untuk melakukan load epoch yang akan di training dan diberikan hasil selama 10 kali epoch. Hasilnya adalah sebagai berikut :

Gambar 7.57 Hasil Soal 12.

13. Soal 13

```
# In[13]:try various model configurations and parameters to find
    the best
# mengimport librari time
import time
#membuat variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1
    fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data
    dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # membuat variabel model Sequential
            model = Sequential()
            #membuat looping untuk variabel i dengan jarak dari
    hasil konvolusi.
            for i in range(conv2d_count):
                # syarat jika i samadengan bobotnya 0
                if i == 0:
                    # menambahkan method add pada variabel model
    dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat
    kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data
    shape yang di hitung dari data train.
                    model.add(Conv2D(32, kernel_size=(3, 3),
    activation='relu', input_shape=np.shape(train_input[0])))
```

```
                    # jika tidak
                else:
                    # menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                # menambahkan method add pada variabel model dengan isian method  Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
                model.add(MaxPooling2D(pool_size=(2, 2)))
            # merubah feature gambar menjadi 1 dimensi vektor
            model.add(Flatten())
            # menambahkan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
            model.add(Dense(dense_size, activation='tanh'))
            # membuat ketentuan jika pemangkasan lebih besar dari 0 persen
            if dropout > 0.0:
                # menambahkan method dropout pada model dengan nilai dari dropout
                model.add(Dropout(dropout))
                # menambahkan method dense dengan fungsi num classs dan rumus softmax
            model.add(Dense(num_classes, activation='softmax'))
            # mongkompile variabel model dengan hasi loss optimasi dan akurasi matrix
            model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
            # melakukan log pada dir
            log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
            # membuat variabel tensorboard dengan isian dari librari keras dan nilai dari lig dir
            tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
            # membuat variabel start dengan isian dari librari time menggunakan method time

            start = time.time()
            # menambahkan method fit pada model dengan data dari train input train output nilai batch nilai epoch verbose nilai 20 persen validation split dan callback dengan nilai tnsorboard.
            model.fit(train_input, train_output, batch_size=32, epochs=10,
                      verbose=0, validation_split=0.2, callbacks=[tensorboard])
            # membuat variabel score dengan nilai evaluasi dari model menggunakan data tes input dan tes output
            score = model.evaluate(test_input, test_output, verbose=2)
            # membuat variabel end
            end = time.time()
            # membuat variabel elapsed
            elapsed = end - start
```

```
            # mencetak hasil perhitungan
            print("Conv2D count: %d, Dense size: %d, Dropout: %.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))
```

Kode di atas befungsi untuk melakukan konfigurasi model dengan parameter yang ditentukan dan mencoba mencari data yang terbaik. Hasilnya adalah sebagai berikut :

```
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.16, Accuracy: 0.74, Time: 459 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.92, Accuracy: 0.70, Time: 447 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.82, Accuracy: 0.77, Time: 453 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.70, Accuracy: 0.77, Time: 458 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.38, Accuracy: 0.74, Time: 456 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.12, Accuracy: 0.70, 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: 1.68, Accuracy: 0.74, Time: 460 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.25 - Loss: 1.45, Accuracy: 0.75, Time: 466 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.50 - Loss: 1.24, Accuracy: 0.75, Time: 475 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.75 - Loss: 1.01, Accuracy: 0.77, Time: 478 sec
Conv2D count: 1, Dense size: 1024, Dropout: 0.00 - Loss: 2.13, Accuracy: 0.72, Time: 518 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: 0.96, 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: 2.05, Accuracy: 0.70, Time: 629 sec
Conv2D count: 1, Dense size: 2048, Dropout: 0.50 - Loss: 1.55, Accuracy: 0.72, Time: 631 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: 0.70, Time: 531 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.25 - Loss: 0.79, Accuracy: 0.77, Time: 539 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.50 - Loss: 0.76, Accuracy: 0.78, Time: 568 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.75 - Loss: 0.76, Accuracy: 0.72, Time: 573 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.00 - Loss: 0.99, Accuracy: 0.70, Time: 550 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.25 - Loss: 0.85, Accuracy: 0.77, Time: 565 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: 565 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: 0.96, 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: 1.05, 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: 0.89, Accuracy: 0.76, Time: 630 sec
```

Gambar 7.58 Hasil Soal 13.

14. Soal 14

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

```
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

Kode di atas befungsi untuk membuat data training kembali dengan model yang sudah di tentukan dan mencari yang terbaik dari hasil training tersebut. Hasilnya adalah sebagai berikut :

Gambar 7.59 Hasil Soal 14.

15. Soal 15

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

Kode di atas befungsi untuk melakukan join terhadap data train dan test dengan seluruh konfigurasi yang telah di tentukan sebelumnya. Hasilnya adalah sebagai berikut :

```
In [16]: model.fit(np.concatenate((train_input, test_input)),
    ...:                np.concatenate((train_output, test_output)),
    ...:                batch_size=4, epochs=10, verbose=1)
Epoch 1/10
 - 126s - loss: 1.7795 - accuracy: 0.5871
Epoch 2/10
 - 114s - loss: 1.0744 - accuracy: 0.7074
Epoch 3/10
 - 130s - loss: 0.9564 - accuracy: 0.7320
Epoch 4/10
 - 115s - loss: 0.8977 - accuracy: 0.7458
Epoch 5/10
 - 118s - loss: 0.8573 - accuracy: 0.7527
Epoch 6/10
 - 116s - loss: 0.8297 - accuracy: 0.7586
Epoch 7/10
 - 126s - loss: 0.8075 - accuracy: 0.7632
Epoch 8/10
 - 122s - loss: 0.7917 - accuracy: 0.7663
Epoch 9/10
 - 129s - loss: 0.7765 - accuracy: 0.7707
Epoch 10/10
 - 118s - loss: 0.7637 - accuracy: 0.7718
Out[16]: <keras.callbacks.callbacks.History at 0x21637ebc348>
```

Gambar 7.60 Hasil Soal 15.

16. Soal 16

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

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai mathsymbols.model. Hasilnya adalah sebagai berikut :

```
# save the trained model
model.save("mathsymbols.model")
```

Gambar 7.61 Hasil Soal 16.

17. Soal 17

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

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai classes.npy dengan reverse ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

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

Gambar 7.62 Hasil Soal 17.

18. Soal 18

```
# In[18]:load the pre-trained model and predict the math symbol
    for an arbitrary image;
# the code below could be placed in a separate file
# mengimpport librari keras model
import keras.models
```

```
# membuat variabel model2 untuk meload model yang telah di simpan tadi
model2 = keras.models.load_model("mathsymbols.model")
# mencetak hasil model2
print(model2.summary())
```

Kode di atas befungsi untuk melakukan load data yang sudah di train dan juga memprediksikan hasil. Hasilnya adalah sebagai berikut :

```
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
```

Gambar 7.63 Hasil Soal 18.

19. Soal 19

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

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

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

```
print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

Kode di atas befungsi untuk melakukan restore terhadap nama class pada fungsi integer encoder, dengan membuat fungsi predict. Hasilnya adalah sebagai berikut :

```
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
```

Gambar 7.64 Hasil Soal 19.

20. Soal 20

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

Kode di atas befungsi untuk menunjukkan hasil akhir prediski. Hasilnya adalah sebagai berikut :

```
# grab an image (we'll just use a random training image for demonstration purposes)
predict("HASYv2/hasy-data/v2-00010.png")
Prediction: A, confidence: 0.87
predict("HASYv2/hasy-data/v2-00500.png")
Prediction: \pi, confidence: 0.58
predict("HASYv2/hasy-data/v2-00700.png")
Prediction: \alpha, confidence: 0.88
```

Gambar 7.65 Hasil Soal 20.

7.3.3 Penanganan Error

1. KeyboardInterrupt

```
  File "<ipyn.ipynb-cell-20-f0d7b0d7c7ac>", line 1, in <module>
    img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
  File "C:\Users\ASUS\ PC\Anaconda3\lib\site-packages\PIL\Image.py", line 2809, in open
    fp = builtins.open(filename, "rb")
KeyboardInterrupt
```

Gambar 7.66 KeyboardInterrupt

2. Cara Penanganan Error

   - KeyboardInterrupt
     Error tersebut karena disebabkan oleh s yang melakukan klik pada keyboard saat running.

7.3.4 Bukti Tidak Plagiat

Gambar 7.67 Bukti Tidak Melakukan Plagiat Chapter 7

7.3.5 Link Video Youtube

https://youtu.be/pV9yrZNEZj4

7.4 1174079 - Chandra Kirana Poetra

7.4.1 Teori

7.4.1.1 Kenapa file teks harus di lakukan tokenizer

Karena tokenizer berguna untuk memproses atau memisahkan bagian bagian pada teks menjadi beberapa bagian, seperti kalimat atau kata. tokenizer bekerja dengan cara memisahkan kata berdasarkan spasi dan tanda baca

Gambar 7.68 Teori 1

7.4.1.2 konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1.

Konsep sederhana dari K Fold Cross Validation ialah Pada code ini:

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

Kfold diatas bertujuan untuk membagi data kedalam 5 bagian yang nantinya akan menampung komentar dari youtube. data yang dibagi ini nanti akan dihitung dan akan menghasilkan presentase yang bisa dibilang cukup baik

Gambar 7.69 Teori 2

7.4.1.3 Apa maksudnya kode program for train, test in splits

Data for train adalah data training set, data ini adalah data yang akan kita uji, sementara test in splits adalah data yang akan kita gunakan untuk validasi data training set, keduanya kemudian dibandingkan dan menghasilkan suatu presentase.

Gambar 7.70 Teori 3

7.4.1.4 Apa maksudnya kode program train content = d[â^CONTENTâ^].iloc[train idx] dan test content = d[â^CONTENTâ^].iloc[test idx]

Fungsi dalam kode tersebut berfungsi untuk mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Contoh sederhananya ketika data telah diubah menjadi data train dan data test maka kita dapat memilihnya untuk ditampilkan pada kolom yang di inginkan.

```
import pandas as pd

mydict = [{'a': 1, 'b': 2, 'c': 3, 'd': 4},
          {'a': 100, 'b': 200, 'c': 300, 'd': 400},
          {'a': 1000, 'b': 2000, 'c': 3000, 'd': 4000 }]
df = pd.DataFrame(mydict)
df
```

```
Out[2]:
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Gambar 7.71 Teori 4

7.4.1.5 Apa maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content)

Berfungsi untuk melakukan proses vektorisasi data menjadi token sebanyak 2000 kata

Gambar 7.72 Teori 5

7.4.1.6 Apa maksud dari fungsi d train inputs = tokenizer.texts to matrix(train content, mode=â^tfidf â^) dan d test inputs = tokenizer.texts to matrix(test content, mode=â^tfidf â^)

Memasukan teks kedalam suatu matrix dengan menggunakan metode TFIDF dan memasukan data testing yang akan diterjemahkan ke suatu matriks

Gambar 7.73 Teori 6

7.4.1.7 Apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train inputs)) dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs))

Fungsi np.amax digunakan untuk mencari nilai maksimal dari suatu array Jika array tidak ada sumbunya, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi a.ndim - 1.

```
>>> a = np.arange(4).reshape((2,2))
>>> a
array([[0, 1],
       [2, 3]])
>>> np.amax(a)           # Maximum of the flattened array
3
>>> np.amax(a, axis=0)   # Maxima along the first axis
array([2, 3])
>>> np.amax(a, axis=1)   # Maxima along the second axis
array([1, 3])
>>> np.amax(a, where=[False, True], initial=-1, axis=0)
array([-1,  3])
>>> b = np.arange(5, dtype=float)
>>> b[2] = np.NaN
>>> np.amax(b)
nan
>>> np.amax(b, where=~np.isnan(b), initial=-1)
4.0
>>> np.nanmax(b)
4.0
```

Gambar 7.74 Teori 7

7.4.1.8 Apa maksud fungsi dari d train outputs = np utils.to categorical(d[â^CLASSâ^].ilo idx]) dan d test outputs = np utils.to categorical(d[â^CLASSâ^].iloc[test idx]) dalam kode program

Membuat variable dengan nama train outputs yang diisi dengan kategori dari class dengan menggunakan ketentuan iloc train idx. Kemudian menampungnya.

Gambar 7.75 Teori 8

7.4.1.9 Apa maksud dari fungsi di listing 7.2.

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

Fungsinya adalah model perlu mengetahui jenis data input apa yang digunakan atau yang seharusnya. Oleh karena itu, lapisan pertama adalah model Sequential (Karena model sequential ini dapat mengerjakan inferensi secara otomatis) perlu menerima informasi tentang bentuk inputnya.

Gambar 7.76 Teori 9

7.4.1.10 Apa maksud dari fungsi di listing 7.3 dengan parameter tersebut

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

Kode diatas merupakan fungsi yang digunakan untuk mendefinisikan loss function, optimizer, dan juga metrics yang akan digunakan pada data yang akan kita train. categoricalcrossentropy merupakan function loss yang kita gunakan untuk melakukan perhitungan, sementara optimizer yang kita gunakan adalah adam, selain adam, ada

juga SGD,RMSprop, dan lain lain dan metrics accuracy adalah value yang akan kita cari nilainya

Gambar 7.77 Teori 10

7.4.1.11 Apa itu Deep Learning

Merupakan cabang dari machine learning yang berasal dari cabang lainnya yaitu artificial intelligence, deep learning mempunyai kemampuan jaringan pembelajaran dengan metode unsupervised dari data yang tidak ada strukturnya maupun tidak ada label yang dikenal juga sebagai deep neural network

7.4.1.12 Apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning

Komponen Neural netwrok adalah neuron yang dilabeli sebagai j yang menerima input dari neuron sebelumnya, seringkali dalam bentuk fungsi identifikasi untuk menyediakan output, sementara deep learning menggunakan motherboard, processor, ram, dan cpu untuk melakukan prosesnya

Neural netwrok menggunakan metode feed forward neural network atau bisa juga recurent network sementara deep learning menggunakan unsupervised pretrained network atau convulutional neural network

7.4.1.13 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.

Algoritma Konvolusi merupakan suatu algoritma yang dapat memproses gambar yang nantinya digunakan untuk membedakan satu gambar dengan yang lainnya, strides merupakan salah satu parameter yang ada, yang digunakan untuk mendefinisikan jumlah pergerakan pixel pada input matrix, ketika didefinisikan 1 stridenya, maka satu pixels akan pindah dalam satu waktu, jika 2 maka dua pixel akan pindah secara bersamaan dalam satu waktu

Gambar 7.78 Teori 11

7.4.2 Praktek

7.4.2.1 Nomor 1

```
import csv #Import library csv
from PIL import Image as pil_image #Import library Image yaitu fungsi PIL (Python Imaging Library) yang berguna untuk mengolah data berupa gambar
import keras.preprocessing.image #Import library keras yang menggunakan method preprocessing yang digunakan untuk membuat neural network
```

7.4.2.2 Nomor 2

```
imgs = [] #Membuat variabel imgs
classes = [] #Membuat variabel classes dengan variabel kosong
with open('E:/hasy-data-labels.csv') as csvfile: #Membuka file hasy-data-labels.csv
    csvreader = csv.reader(csvfile) #Membuat variabel csvreader yang berisi method csv.reader untuk membaca csvfile
    i = 0 # membuat variabel i dengan isi 0
    for row in csvreader: # Membuat looping pada variabel csvreader
        if i > 0: #Ketentuannya jika i lebih kecil daripada 0
            img = keras.preprocessing.image.img_to_array(pil_image.open("E:/HASYv2/" + row[0])) #Dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            # 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 #Membagi data yang ada pada fungsi img sebanyak 255.0
            imgs.append((row[0], row[2], img)) #Menambah nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            classes.append(row[2]) #Menambahkan nilai pada row ke 2 pada variabel classes
        i += 1 #Menambah nilai satu pada variabel i
```

7.4.2.3 Nomor 3

```
import random #Mengimport library random
random.shuffle(imgs) #Melakukan randomize pada fungsi imgs
```

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

7.4.2.4 Nomor 4

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

7.4.2.5 Nomor 5

```
from sklearn.preprocessing import LabelEncoder #Mengimport library LabelEncode dari sklearn
from sklearn.preprocessing import OneHotEncoder #Mengimport library OneHotEncoder dari sklearn
```

7.4.2.6 Nomor 6

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

7.4.2.7 Nomor 7

```
onehot_encoder = OneHotEncoder(sparse=False)#Membuat variabel onehot_encoder dengan isi OneHotEncoder
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1) #Mengisi variabel integer_encoded dengan isi integer_encoded yang telah di convert pada fungsi sebelumnya
onehot_encoder.fit(integer_encoded) #Mengkonvert variabel integer_encoded kedalam onehot_encoder
```

7.4.2.8 Nomor 8

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

7.4.2.9 Nomor 9

```
from keras.models import Sequential #Mengimport library Sequential dari Keras
from keras.layers import Dense, Dropout, Flatten #Mengimport library Dense, Dropout, Flatten dari Keras
from keras.layers import Conv2D, MaxPooling2D #Mengimport library Conv2D, MaxPooling2D dari Keras
```

7.4.2.10 Nomor 10

```
model = Sequential() #Membuat variabel model dengan isian library Sequential
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0]))) #Variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(MaxPooling2D(pool_size=(2, 2))) #Variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixel
model.add(Conv2D(32, (3, 3), activation='relu')) #Variabel model di tambahkan dengan library Conv2D 32bit dengan kernel 3 x 3
model.add(MaxPooling2D(pool_size=(2, 2))) #Variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(Flatten()) #Variabel model di tambahkan library Flatten
model.add(Dense(1024, activation='tanh')) #Variabel model di tambahkan library Dense dengan fungsi tanh
model.add(Dropout(0.5)) #Variabel model di tambahkan library dropout untuk memangkas data tree sebesar 50 persen
model.add(Dense(num_classes, activation='softmax')) #Variabel model di tambahkan library Dense dengan data dari num_classes dan fungsi softmax
```

```
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy']) #Mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
print(model.summary()) #Mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
```

7.4.2.11 Nomor 11

```
import keras.callbacks #Mengimport library keras dengan fungsi callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='E:/KB/hasyv2/logs/mnist-style') #Membuat variabel tensorboard dengan isi lib keras
```

7.4.2.12 Nomor 12

```
model.fit(train_input, train_output, #Fungsi model ditambahkan fungsi fit untuk mengetahui perhitungan dari train_input train_output
          batch_size=32, #Dengan batch size 32 bit
          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])
```

7.4.2.13 Nomor 13

```
import time #Mengimport library time
results = [] #Membuat variabel result dengan array kosong
for conv2d_count in [1, 2]: #Melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
    for dense_size in [128, 256, 512, 1024, 2048]: #Menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
        for dropout in [0.0, 0.25, 0.50, 0.75]: #Membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
            model = Sequential() #Membuat variabel model Sequential
            for i in range(conv2d_count): #Membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
                if i == 0: #Syarat jika i samadengan bobotnya 0
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0]))) #Menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                else: #Jika tidak
```

```
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu')) #Menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                model.add(MaxPooling2D(pool_size=(2, 2))) #Menambahkan method add pada variabel model dengan isian method Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
            model.add(Flatten()) #Merubah feature gambar menjadi 1 dimensi vektor
            model.add(Dense(dense_size, activation='tanh')) #Menambahkan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
            if dropout > 0.0: #Membuat ketentuan jika pemangkasan lebih besar dari 0 persen
                model.add(Dropout(dropout)) #Menambahkan method dropout pada model dengan nilai dari dropout
            model.add(Dense(num_classes, activation='softmax')) #Menambahkan method dense dengan fungsi num classs dan rumus softmax
            model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy']) #Mengcompile variabel model dengan hasi loss optimasi dan akurasi matrix
            log_dir = 'E:/KB/hasyv2/logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout) #Melakukan log
            tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir) # membuat variabel tensorboard dengan isian dari library keras dan nilai dari log_dir

            start = time.time() #Membuat variabel start dengan isian dari library time menggunakan method time
            model.fit(train_input, train_output, batch_size=32, epochs=10,
                      verbose=0, validation_split=0.2, callbacks=[tensorboard]) #Menambahkan method fit pada model dengan data dari train input train output nilai batch nilai epoch verbose nilai 20 persen validation split dan callback dengan nilai tensorboard.
            score = model.evaluate(test_input, test_output, verbose=2) #Membuat variabel score dengan nilai evaluasi dari model menggunakan data tes input dan tes output
            end = time.time() #Membuat variabel end
            elapsed = end - start  #Membuat variabel elapsed
            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)) #Mencetak hasil perhitungan
            results.append((conv2d_count, dense_size, dropout, score[0], score[1], elapsed))
```

7.4.2.14 Nomor 14

```
model = Sequential() #Membuat variabel model dengan isian library Sequential
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))#Variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3
```

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

7.4.2.15 Nomor 15

```
model.fit(np.concatenate((train_input, test_input)), #Melakukan
    training yang isi datanya dari join numpy menggunakan data
    train_input test_input
          np.concatenate((train_output, test_output)), #Kelanjutan
    data yang di gunakan pada join train_output test_output
          batch_size=32, epochs=10, verbose=2) #Menggunakan ukuran 32
     bit dan epoch 10
```

7.4.2.16 Nomor 16

```
model.save("mathsymbols.model") #Menyimpan model atau mengeksport
    model yang telah di jalan tadi
```

7.4.2.17 Nomor 17

```
np.save('classes.npy', label_encoder.classes_) #Menyimpan label
    encoder dengan nama classes.npy
```

7.4.2.18 Nomor 18

```
import keras.models #Mengimpport library keras model
model2 = keras.models.load_model("mathsymbols.model") #Membuat
    variabel model2 untuk meload model yang telah di simpan tadi
print(model2.summary()) #Mencetak hasil model2
```

7.4.2.19 Nomor 19

```
label_encoder2 = LabelEncoder() # membuat variabel label encoder ke 2 dengan isian fungsi label encoder.
label_encoder2.classes_ = np.load('classes.npy') #Menambahkan method classess dengan data classess yang di eksport tadi
def predict(img_path): #Membuat fumgsi predict dengan path img
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path)) #Membuat variabel newimg dengam membuay immage menjadi array dan membuka data berdasarkan img path
    newimg /= 255.0 #Membagi data yang terdapat pada variabel newimg sebanyak 255

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

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

7.4.2.20 Nomor 20

```
predict("HASYv2/hasy-data/v2-00010.png") #Mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00010.png
predict("HASYv2/hasy-data/v2-00500.png") #Mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00500.png
predict("HASYv2/hasy-data/v2-00700.png") #Mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00700.png
```

7.4.3 Penanganan Error

7.4.3.1 Error

- NameError

```
NameError: name 'np' is not defined
```

Gambar 7.79 NameError

7.4.3.2 Solusi Error

- NameError

  Pastikan sudah diimport atau cek typo

7.4.4 Bukti Tidak Plagiat

Gambar 7.80 Bukti tidak plagiat

7.4.5 Link Youtube

https://youtu.be/AHJOIZJYd9I

7.5 Muhammad Reza Syachrani - 1174084

7.5.1 Teori

1. Jelaskan kenapa le teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar
   untuk mempermudah mesin dalam mengolah file teks yang dimana bentuk teks tersebut di konversi menjadi urutan integer kata atau vector binary. ilustrasi tokenizer sebagai berikut, saya memiliki teks yaitu ”saya makan nasi goreng” dan setelah dilakukan tokenizer berubah menjadi ’saya’: 1, ’makan’: 2, ’nasi’: 3, ’goreng’: 4

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube.
   pada codingan tersebut terdapat variabel kfold yang berisi fungsi StratifiedK-Fold yang memiliki parameter n_splits=5 yang berarti melakukan pengulangan terhadap data masing-masing lima kali dengan atribut class sebagai acuan pengolahan datanya kemudian akan di hasilkan akurasi dari pengulangan data tersebut sebesar sekian persen tergantung datanya.

Gambar 1 – Skema 10 fold CV

Gambar 7.81 Teori 2

3. Jelaskan apa maksudnya kode program for train, test in splits.dilengkapi dengan ilustrasi atau gambar.
   kode program for train fungsinya untuk melakukan training terhadap data yang telah di deklarasikan sebelumnya. sedangkan kode program test in split fungsinya untuk membatasi jumlah data yang akan digunakan.

```
import numpy as np
from sklearn.model_selection import train_test_split
x, y = np.arange(25).reshape((5, 5)), range(5)
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.2, random_state=5252)
x_train, x_test
```

```
(array([[20, 21, 22, 23, 24],
        [10, 11, 12, 13, 14],
        [15, 16, 17, 18, 19],
        [ 0,  1,  2,  3,  4]]), array([[5, 6, 7, 8, 9]]))
```

Gambar 7.82 Teori 3

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 tersebut adalah membaca isian kolom pada field yang bernama CONTENT sebagai data training sedangkan kode program test content membaca isi kolom pada field yang bernama CONTENT sebagai data testing.

Index	CONTENT
0	i love this so much. AND als…
1	http:// www.billboard…
2	Hey guys! Please join m…
3	http:// psnboss.com/?…
4	Hey everyone. Watch this tr…
5	check out my rapping hope …

Gambar 7.83 Teori 4

5. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num_words=2000) dan tokenizer.t_on_texts(train_content), dilengkapi dengan ilustrasi atau gambar.
   fungsi tokenizer = Tokennizer(num_words=2000) untuk membaca kalimat menjadi token/indeks sebanyak 2000 kata dan fungsi fit_on_texts(train_konten) un-

tuk membuat membaca data token/indeks teks yang telah di masukan kedalam data train_konten.

Gambar 7.84 Teori 5

6. Jelaskan apa maksud dari fungsi d_train_inputs = tokenizer.texts_to_matrix(train_content, mode=tdf) dan d_test_inputs = tokenizer.texts_to_matrix(test_content, mode=tdf), dilengkapi dengan ilustrasi kode dan atau gambar
variabel d_train_inputs melakukan tokenizer dari bentuk teks menjadi bentuk matrix yang berurutan dari data train_content dengan mode tf idf begitu juga dengan d_test_inputs untuk data test.

Fungsi **texts_to_matrix()** dapat digunakan untuk membuat satu vektor per dokumen, menyediakan skema standar teks encoding bag-of-words melalui mode-mode argumen dari fungsi, yaitu :

- **binary** : apakah setiap kata ada dalam dokumen atau tidak (default)
- **count** : jumlah setiap kata dalam dokumen
- **tfidf** : skor untuk Text Frequency-Inverse DocumentFrequency (TF-IDF) untuk setiap kata dalam dokumen
- **freq** : frekuensi setiap kata sebagai rasio kata dalam setiap dokumen.

Gambar 7.85 Teori 6

7. Jelaskan apa maksud dari fungsi d_train_inputs = d_train_inputs/np.amax(np.absolute(d_tra dan d_test_inputs = d_test_inputs/np.amax(np.absolute(d_test_inputs)), dilengkapi dengan ilustrasi atau gambar.
fungsi tersebut digunakan untuk membagi matrix tfidf untuk menentukan maksimum array yang kemudian hasilnya tersebut dimasukan ke dalam variabel d_train_input dan d_test_input dengan metode absolute sehingga tanpa adanya bilangan negatif dan nol.

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

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

Gambar 7.86 Teori 7

8. Jelaskan apa maksud fungsi dari d_train_outputs = np_utils.to_categorical(d[CLASS].iloc[t dan d_test_outputs = np_utils.to_categorical(d[CLASS].iloc[test_idx]) dalam kode program, dilengkapi dengan ilustrasi atau gambar.
   fungsi untuk melakukan one-hot encoding merubah nilai vektor dengan bentuk integer yang ada pada atribut class menjadi bentuk matrix biner untuk atribut CLASS sehingga hanya ada dua pilihan yaitu 0 atau 1.

```
# 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 7.87 Teori 8

9. Jelaskan apa maksud dari fungsi di listing 7.2. Gambarkan ilustrasi Neural Network nya dari model kode tersebut.
   fungsi kode tersebut untuk melakukan permodelan sequential yang digunakan untuk mencari data dengan menerima parameter atau argumen kunci. kemudian di tambhakan metod add dengan danse sebanyak 512 neuron inputan dengan input shape 2000 vektor yang sudah dinormalisasi. kemudian di tambahkan lagi fungsi aktivasi dengan fungsi ”relu”. dan dilakukan overfitting atau pemotongan

bobot sebesar 0.5 atau 50 persen. Lalu pada layer output terdapat 2 neuron output. Kemudian outputan tersebut diaktivasi menggunakan fungsi softmax.

Gambar 7.88 Teori 9

10. Jelaskan apa maksud dari fungsi di listing 7.3 dengan parameter tersebut.
    fungsi kode yang melakukan compile pada model dengan menggunakan beberapa parameter seperti loss yang akan mengembalikan fungsi nilai loss yang diambil, sedangkan fungsi adamax digunakan untuk mengetahui nilai lossnya, dan matrics = akurasi merupakan akurasi dari nilai matriknya.

11. Jelaskan apa itu Deep Learning
    Merupakan metode pada machine learning yang menggunakan artificial neural networks. algoritma permodelan 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 merupakan algoritma jaringan syaraf yang akan melakukan pembobotan terhadap data yang sudah ada sebagai acuan untuk data inputan selanjutnya. kemudian terdiri atas beberapa lapisan/layar atau hiden layer. perbedaan antara deep learning dan Deep Neural Network yaitu Deep Neural Network merupakan algoritma sedangkan deep learning yang menggunakan Deep Neural Network tersebut.

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
    Karena NPM saya 1164084 dan hasil dari (NPM mod 3)+1 = 2, maka saya menggunaan stride 2 dengan ketentuan max pooling 2x2.

Gambar 7.89 Teori 13

7.5.2 Praktek

1. Jelaskan kode program pada blok In[1]

```
# In[1]:import lib
import csv
#Mengimport library csv untuk mengimport file ekstensi .csv
from PIL import Image as pil_image
#Mengimport library Image dari PIL sebagai pil_image yang
    digunakan untuk mengolah data gambar
import keras.preprocessing.image
#Mengimport library keras dengan metode preprocessing.image yang
    digunakan untuk membuat neural network
```

```
In [1]: import csv
   ...: #Mengimport library csv untuk mengimport file ekstensi .csv
   ...: from PIL import Image as pil_image
   ...: #Mengimport library Image dari PIL sebagai pil_image yang digunakan untuk
mengolah data gambar
   ...: import keras.preprocessing.image
   ...: #Mengimport library keras dengan metode preprocessing.image yang digunakan
untuk membuat neural network
Using TensorFlow backend.
```

Gambar 7.90 Hasil Praktek 1

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

```
# In[2]:load all images (as numpy arrays) and save their classes

imgs = []
#Membuat variabel imgs dengan variabel kosong
classes = []
#Membuat variabel imgs dengan variabel kosong
with open('D:/New folder/KB3C/src/1174084/7/HASYv2/hasy-data-
    labels.csv') as csvfile:
    #membuka file csv pada folder HASYv2 yaitu hasy-data-labels.
    csv sebagai csvfile
    csvreader = csv.reader(csvfile)
    #membuat variabel csvreader yang berisikan metode reader dari
     library csv yang membaca csvfile.
    i = 0
    #Membuat varianel i yang berisikan 0
```

```
    for row in csvreader:
    #Membuat pengulangan pada variabel csvreader
        if i > 0:
        #Dengan ketentuan jika i lebih besar dari 0
            img = keras.preprocessing.image.img_to_array(pil_image.open("D:/New folder/KB3C/src/1174084/7/HASYv2/" + row[0]))
            #Membuat variabel img yang berisikan fungsi keras untuk aktivasi neural network yang membaca data pada folder HASYv2 yang dibuka dengan row berparameter 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
            #Membagi data yang berada pada variabel img dengan 255.0
            imgs.append((row[0], row[2], img))
            #Menambahkan nilai baru pada imgs yaitu row 0,row 2 dilanjutkan dengan variabel img.
            classes.append(row[2])
            # menambahkan nilai pada row ke 2 pada variabel classes
        i += 1
        #Menambahkan nilai 1 pada variabel i
```

Name	Type	Size	Value
classes	list	168233	['A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', ...]
i	int	1	168234
img	float32	(32, 32, 3)	[[[1. 1. 1.] [1. 1. 1.]
imgs	list	168233	[('hasy-data/v2-00000.png', 'A', Numpy array), ('hasy-data/v2-00001.pn ...
row	list	4	['hasy-data/v2-168232.png', '1400', '\guillemotleft', '16925']

Gambar 7.91 Hasil Praktek 2

3. Jelaskan kode program pada blok In[3]

```
# In[3]:shuffle the data, split into 80% train, 20% test

import random
#Mengimport library random
random.shuffle(imgs)
#Melakukan shuffle pada variabel imgs
split_idx = int(0.8*len(imgs))
#Membuat variabel split_idx yang diisi dengan nilai integer dari perkaslian 80% dengan jumlah dari variabel imgs
train = imgs[:split_idx]
#Membuat variabel train yang diisi dengan dengan pemecahan index awal pada data variabel split_idx
test = imgs[split_idx:]
#Membuat variabel test yang diisi dengan pemecahan index akhir pada data variabel split_idx
```

split_idx	int	1	134586
test	list	33647	[('hasy-data/v2-135728.png', '\vdots', Numpy array), ('hasy-data/v2-84 ...
train	list	134586	[('hasy-data/v2-165563.png', '\sun', Numpy array), ('hasy-data/v2-0377 ...

Gambar 7.92 Hasil Praktek 3

4. Jelaskan kode program pada blok In[4]

```
# In[4]:

import numpy as np
#Mengimport library numpy sebagai np
train_input = np.asarray(list(map(lambda row: row[2], train)))
#Membuat variabel train_input dengan np method asarray yang mana membuat array dangan fungsi list yang didalamnya diterapkan fungsi map untuk mengembalikan interator
#dan menggunakan lamba untuk mengecilkan fungsi dari objek yang berada pada row 2 dari data train
test_input = np.asarray(list(map(lambda row: row[2], test)))
#Membuat variabel test_input dengan isi np method asarray yang mana membuat array dangan fungsi list yang didalamnya diterapkan fungsi map untuk mengembalikan interator
#dan menggunakan lamba untuk mengecilkan fungsi dari objek yang berada pada row 2 dari data test
train_output = np.asarray(list(map(lambda row: row[1], train)))
#Membuat variabel train_output dengan np method asarray yang mana membuat array dangan fungsi list yang didalamnya diterapkan fungsi map untuk mengembalikan interator
#dan menggunakan lamba untuk mengecilkan fungsi dari objek yang berada pada row 1 dari data train
test_output = np.asarray(list(map(lambda row: row[1], test)))
#Membuat variabel test_output dengan np method asarray yang mana membuat array dangan fungsi list yang didalamnya diterapkan fungsi map untuk mengembalikan interator
#dan menggunakan lamba untuk mengecilkan fungsi dari objek yang berada pada row 1 dari data train
```

test_input	float32	(33647, 32, 32, 3)	[[[[1. 1. 1.] [1. 1. 1.]
test_output	str608	(33647,)	ndarray object of numpy module
train	list	134586	[('hasy-data/v2-165563.png', '\sun', Numpy array), ('hasy-data/v2-0377 ...
train_input	float32	(134586, 32, 32, 3)	[[[[1. 1. 1.] [1. 1. 1.]
train_output	str608	(134586,)	ndarray object of numpy module

Gambar 7.93 Hasil Praktek 4

5. Jelaskan kode program pada blok In[5]

```
# In[5]: import encoder and one hot
from sklearn.preprocessing import LabelEncoder
```

```
#Mengimport library LabelEncoder dari sklearn.preprocessing yang digunakan untuk mengonversi jenis data teks kategori menjadi data numerik
from sklearn.preprocessing import OneHotEncoder
#Mengimport library OneHotEncoder dari sklearn.preprocessing
```

```
In [7]: from sklearn.preprocessing import LabelEncoder
   ...: #Mengimport library LabelEncoder dari sklearn.preprocessing yang digunakan
untuk mengonversi jenis data teks kategori menjadi data numerik
   ...: from sklearn.preprocessing import OneHotEncoder
   ...: #Mengimport library OneHotEncoder dari sklearn.preprocessing
```

Gambar 7.94 Hasil Praktek 5

6. Jelaskan kode program pada blok In[6]

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

# first, convert class names into integers
label_encoder = LabelEncoder()
#Membuat variabel label_encoder dengan isi LabelEncoder
integer_encoded = label_encoder.fit_transform(classes)
#Membuat variabel integer_encoded yang berfungsi untuk mengkonversi variabel classes kedalam bentuk integer
```

integer_encoded	int64	(168233,)	[15 15 15 ... 143 143 143]

Gambar 7.95 Hasil Praktek 6

7. Jelaskan kode program pada blok In[7]

```
# In[7]:then convert integers into one-hot encoding
onehot_encoder = OneHotEncoder(sparse=False)
#Membuat variabel onehot_encoder dengan isi fungsi OneHotEncoder parameter sparse=false
integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
#Membuat variabel integer_encoder dengan isi fungsi integer_encoded yang telah di convert pada fungsi sebelumnya
onehot_encoder.fit(integer_encoded)
#Onehotencoding melakukan fitting pada variabel integer_encoded
```

integer_encoded	int64	(168233, 1)	[[15] [15]

Gambar 7.96 Hasil Praktek 7

8. Jelaskan kode program pada blok In[8]

```
# In[8]:convert train and test output to one-hot
train_output_int = label_encoder.transform(train_output)
```

```
#Membuat variabel train_output_int dengan isi hasil konversi data train_output menggunakan label_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
#Mengkonversi data train_output_int menggunakan fungsi onehot_encoder
test_output_int = label_encoder.transform(test_output)
#Membuat variabel test_output_int dengan isi hasil konversi data test_output menggunakan label_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
#Mengkonversi data test_output_int menggunakan fungsi onehot_encoder
num_classes = len(label_encoder.classes_)
#Membuat variabel num_classes dengan isi jumlah class pada label_encoder
print("Number of classes: %d" % num_classes)
#Menampilkan hasil dari variabel num_classes
```

```
In [10]: train_output_int = label_encoder.transform(train_output)
    ...: #Membuat variabel train_output_int dengan isi hasil konversi data
train_output menggunakan label_encoder
    ...: train_output =
onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
    ...: #Mengkonversi data train_output_int menggunakan fungsi onehot_encoder
    ...: test_output_int = label_encoder.transform(test_output)
    ...: #Membuat variabel test_output_int dengan isi hasil konversi data test_output
menggunakan label_encoder
    ...: test_output =
onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
    ...: #Mengkonversi data test_output_int menggunakan fungsi onehot_encoder
    ...: num_classes = len(label_encoder.classes_)
    ...: #Membuat variabel num_classes dengan isi jumlah class pada label_encoder
    ...: print("Number of classes: %d" % num_classes)
    ...: #Menampilkan hasil dari variabel num_classes
Number of classes: 369
```

Gambar 7.97 Hasil Praktek 8

9. Jelaskan kode program pada blok In[9]

```
# In[9]: import sequential
import tensorflow as tf
from keras.models import Sequential
#Mengimport Sequential dari library keras
from keras.layers import Dense, Dropout, Flatten
#Mengimport Dense, Dropout, Flatten dari Library keras
from keras.layers import Conv2D, MaxPooling2D
#Mengimport Conv2D dan MaxPoolinf2D dari library Keras
```

```
In [11]: from keras.models import Sequential
    ...: #Mengimport Sequential dari library keras
    ...: from keras.layers import Dense, Dropout, Flatten
    ...: #Mengimport Dense, Dropout, Flatten dari Library keras
    ...: from keras.layers import Conv2D, MaxPooling2D
    ...: #Mengimport Conv2D dan MaxPoolinf2D dari library Keras
```

Gambar 7.98 Hasil Praktek 9

10. Jelaskan kode program pada blok In[10]

```
# In[10]: desain jaringan
model = tf.keras.Sequential()
#Membuat variabel model dengan isi fungsi Sequential
model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3),
    activation='relu',
                 input_shape=np.shape(train_input[0])))
#Variabel model ditambahkan fungsi Conv2d dengan paramater 32
    filter dengan karnel berukuran 3x3
#dengan algoritam activation relu mengunakan data train_input
    mulai dari baris nol.
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
#Variabel madel ditambahkan fungsi MaxPooling2D dengan ketentuan
    ukuran 2x2
model.add(tf.keras.layers.Conv2D(32, (3, 3), activation='relu'))
#Variabel model ditambahkan fungsi Conv2D dengan 32 filter dengan
     konvolusi berukuran 3x3, menggunakan algoritam activation
    relu
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
#Variabel madel ditambahkan fungsi MaxPooling2D dengan ketentuan
    ukuran 2x2
model.add(tf.keras.layers.Flatten())
#Variabel model di tambahkan fungsi Flatten
model.add(tf.keras.layers.Dense(1024, activation='tanh'))
#Variabel model ditambahakan fungsi dense dengan 1024 neuron, dan
     menggunakan algoritma tanh untuk activation
model.add(tf.keras.layers.Dropout(0.5))
#Variabel model di tambahkan fungsi Dropout sebesar 50% untuk
    mencegah terjadinya overfitting
model.add(tf.keras.layers.Dense(num_classes, activation='softmax'
    ))
#Variabel model ditambahkan fungsi Dense dengan parameter
    variabel num_classes menggunakan activation softmax
model.compile(loss='categorical_crossentropy', optimizer='adam',
               metrics=['accuracy'])
#Variabel model di compile dengan parameter loss, matrik, dan
    optimasi
print(model.summary())
#Menampilkan model yang telah dibuat.
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (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 7.99 Hasil Praktek 10

11. Jelaskan kode program pada blok In[11]

```
# In[11]: import sequential

import keras.callbacks
#mengimport library keras callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='.\\logs\\mnist
    -style')
#Membuat variabel tensorboard dengan isi fungsi TenserBoard yang
    ada pada library keras.callback dengan parameter director log
     './logs/mnist-style'
```

```
In [13]: import keras.callbacks
    ...: #mengimport library keras callbacks
    ...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-style')
    ...: #Membuat variabel tensorboard dengan isi fungsi TenserBoard yang ada pada
library keras.callback dengan parameter director log './logs/mnist-style'
```

Gambar 7.100 Hasil Praktek 11

12. Jelaskan kode program pada blok In[12]

```
# In[12]: 5menit kali 10 epoch = 50 menit
model.fit(train_input, train_output,
#Melakaukan fitting pada model dengan paramater train_input,
    train_output
        batch_size=32,
        #dengan mengunakan batch_size sebesar 32,
        epochs=10,
        #epoche=10 yang berarti terajadi perulanagan sebanyak
    10 kali
        verbose=2,
        #untuk menghasilkan informasi logging dari data yang
    ditentukan dengan nilai 2
```

```
            validation_split=0.2,
            #melakukan pemecahan nilai sebesar 0.2 / 20% dari perhitungan validasi
            callbacks=[tensorboard])
            #mengeksekusi tensorboard dimana digunakan untuk visualisasikan parameter training, metrik, hiperparameter pada nilai/data yang diproses

score = model.evaluate(test_input, test_output, verbose=2)
#Membuat variabel score dengan isi fungsi evuleate dari model dengan paramater test_input, test_output dan verbose=2
#untuk memprediksi output dan input
print('Test loss:', score[0])
#Menampilkan score optimasi dengan ketentuan nilai parameter 0
print('Test accuracy:', score[1])
#Mencetak score akurasi dengan ketentuan nilai parameter 1
```

```
Train on 107668 samples, validate on 26918 samples
Epoch 1/10
107668/107668 - 74s - loss: 1.5624 - accuracy: 0.6235 - val_loss: 1.0111 - val_accuracy: 0.7197
Epoch 2/10
107668/107668 - 75s - loss: 0.9902 - accuracy: 0.7270 - val_loss: 0.9148 - val_accuracy: 0.7464
Epoch 3/10
107668/107668 - 80s - loss: 0.8742 - accuracy: 0.7509 - val_loss: 0.8888 - val_accuracy: 0.7557
Epoch 4/10
107668/107668 - 80s - loss: 0.8059 - accuracy: 0.7650 - val_loss: 0.8823 - val_accuracy: 0.7541
Epoch 5/10
107668/107668 - 82s - loss: 0.7606 - accuracy: 0.7744 - val_loss: 0.8537 - val_accuracy: 0.7570
Epoch 6/10
107668/107668 - 81s - loss: 0.7213 - accuracy: 0.7820 - val_loss: 0.8514 - val_accuracy: 0.7582
Epoch 7/10
107668/107668 - 81s - loss: 0.6873 - accuracy: 0.7899 - val_loss: 0.8616 - val_accuracy: 0.7651
Epoch 8/10
107668/107668 - 81s - loss: 0.6641 - accuracy: 0.7929 - val_loss: 0.8498 - val_accuracy: 0.7615
Epoch 9/10
107668/107668 - 80s - loss: 0.6419 - accuracy: 0.7986 - val_loss: 0.8780 - val_accuracy: 0.7571
Epoch 10/10
107668/107668 - 78s - loss: 0.6204 - accuracy: 0.8029 - val_loss: 0.8908 - val_accuracy: 0.7584
33647/33647 - 6s - loss: 0.8712 - accuracy: 0.7624
Test loss: 0.871180891701912
Test accuracy: 0.76238596
```

Gambar 7.101 Hasil Praktek 12

13. Jelaskan kode program pada blok In[13]

```
# In[13]:try various model configurations and parameters to find the best

import time
#import library time

results = []
#Membuat variabel result dengan isi array kosong
for conv2d_count in [1, 2]:
#Melakukan looping mengunakan convd2d_count dengan ketentuan konvolusi 2 dimensi yaitu 1, 2
    for dense_size in [128, 256, 512, 1024, 2048]:
    #Melakukan looping menggunakan ukuran dari densenya yaitu 123, 256, 512, 1024, 2048.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
        #Melakukan looping menggunakn dropout dengan ketentuan 0%, 25%, 50%, 75% untuk memangkas data.
            model = tf.keras.Sequential()
            #Membuat variabel model dengan isi fungsi sequential.
            for i in range(conv2d_count):
```

```
17          #Melakukan looping menggunakan i dengan jarak hasil konvulasi
18              if i == 0:
19              #jika nilai i sama dengan 0
20                  model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
21                  #Variabel model ditambahkan fungsi Conv2d dengan paramater 32 filter dengan karnel berukuran 3x3
22                  #dengan algoritam activation relu mengunakan data train_input mulai dari baris nol.
23              else:
24              #jika tidak
25                  model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu'))
26                  #Variabel model akan ditambahkan fungsi Conv2d dengan paramater 32 filter dengan karnel berukuran 3x3
27                  #dengan algoritam activation relu tanpa ada parameter input_shape.
28              model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
29              #Variabel madel ditambahkan fungsi MaxPooling2D dengan ketentuan ukuran 2x2
30          model.add(tf.keras.layers.Flatten())
31          #Variabel model di tambahkan fungsi Flatten
32          model.add(tf.keras.layers.Dense(dense_size, activation='tanh'))
33          #Variabel model ditambahakan fungsi dense dengan jumlah dense yang digunakan, dan menggunakan algoritma tanh untuk activation
34          if dropout > 0.0:
35          #jika nilai dari dropout lebih besar dari 0.0
36              model.add(tf.keras.layers.Dropout(dropout))
37              #Variabel model di tambahkan fungsi Dropout sebesar dropout yang digunakan untuk mencegah terjadinya overfitting
38          model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
39          #Variabel model ditambahkan fungsi Dense dengan parameter variabel num_classes menggunakan activation softmax
40          model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
41          #Variabel model di compile dengan parameter loss, matrik, dan optimasi
42          log_dir = '.\\logs\\conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
43          #Melakukan log pada dir
44          tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
45          #Mengisi variabel tensorboard dengan isian dari library keras dan nilai dari log dir
46
47          start = time.time()
48          #Membuat variabel start dengan isi fungsi time dari library time
```

```
        model.fit(train_input, train_output, batch_size=32, epochs=10,
                  verbose=0, validation_split=0.2, callbacks=[tensorboard])
        #Melakukan fitting pada model dengan parameter test_input, test_output, batch_size,epochs,verbose, validation_split dan callbacks
        score = model.evaluate(test_input, test_output, verbose=2)
        #Membuat variabel score dengan nilai evaluasi dari model menggunakan data tes input dan tes output dengan verbose adalah 2
        end = time.time()
        #Membuat variabel end dengan isi fungsi time dari library time
        elapsed = end - start
        #Membuat variabel elapse yang diisi dengan nilai hasil waktu end dikurangi 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))
        #Menampilkan hasil perhitungan.
```

```
33647/33647 - 4s - loss: 1.1405 - accuracy: 0.7386
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.14, Accuracy: 0.74, Time: 462 sec
33647/33647 - 5s - loss: 0.9202 - accuracy: 0.7652
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.92, Accuracy: 0.77, Time: 488 sec
33647/33647 - 6s - loss: 0.8037 - accuracy: 0.7775
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.80, Accuracy: 0.78, Time: 550 sec
WARNING:tensorflow:Large dropout rate: 0.75 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
WARNING:tensorflow:Large dropout rate: 0.75 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
WARNING:tensorflow:Large dropout rate: 0.75 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
instead of keep_prob. Please ensure that this is intended.
33647/33647 - 5s - loss: 0.8012 - accuracy: 0.7706
Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.80, Accuracy: 0.77, Time: 564 sec
33647/33647 - 6s - loss: 1.3329 - accuracy: 0.7401
Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.33, Accuracy: 0.74, Time: 695 sec
33647/33647 - 6s - loss: 1.1072 - accuracy: 0.7603
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.11, Accuracy: 0.76, Time: 738 sec
33647/33647 - 5s - loss: 0.9145 - accuracy: 0.7757
Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.91, Accuracy: 0.78, Time: 731 sec
WARNING:tensorflow:Large dropout rate: 0.75 (>0.5). In TensorFlow 2.x, dropout() uses dropout rate
```

Gambar 7.102 Hasil Praktek 13

14. Jelaskan kode program pada blok In[14]

```
# In[14]:rebuild/retrain a model with the best parameters (from the search) and use all data
model = tf.keras.Sequential()
#Membuat variabel model dengan isi fungsi Sequential
model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
#Variabel model ditambahkan fungsi Conv2d dengan paramater 32 filter dengan karnel berukuran 3x3
#dengan algoritam activation relu mengunakan data train_input mulai dari baris nol.
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
#Variabel madel ditambahkan fungsi MaxPooling2D dengan ketentuan ukuran 2x2
model.add(tf.keras.layers.Conv2D(32, (3, 3), activation='relu'))
```

```
#Variabel model ditambahkan fungsi Conv2D dengan 32 filter dengan konvolusi berukuran 3x3, menggunakan algoritam activation relu
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
#Variabel madel ditambahkan fungsi MaxPooling2D dengan ketentuan ukuran 2x2
model.add(tf.keras.layers.Flatten())
#Variabel model di tambahkan fungsi Flatten
model.add(tf.keras.layers.Dense(128, activation='tanh'))
#Variabel model ditambahakan fungsi dense dengan 128 neuron, dan menggunakan algoritma tanh untuk activation
model.add(tf.keras.layers.Dropout(0.5))
#Variabel model di tambahkan fungsi Dropout sebesar 50% untuk mencegah terjadinya overfitting
model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
#Variabel model ditambahkan fungsi Dense dengan parameter variabel num_classes menggunakan activation softmax
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
#Variabel model di compile dengan parameter loss, matrik, dan optimasi
print(model.summary())
#Menampilkan ringkasan model yang telah dibuat.
```

```
Model: "sequential_19"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_24 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_23 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_25 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_24 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
flatten_19 (Flatten)         (None, 1152)              0
_________________________________________________________________
dense_36 (Dense)             (None, 128)               147584
_________________________________________________________________
dropout_13 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_37 (Dense)             (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.103 Hasil Praktek 14

15. Jelaskan kode program pada blok In[15]

```
# In[15]:join train and test data so we train the network on all data we have available to us
model.fit(np.concatenate((train_input, test_input)),
#Melakukan fitting pada model melakukan join numpy menggunakan data train_input test_input
          np.concatenate((train_output, test_output)),
```

```
        #kemudian join numpy menggunakan data train_output test_output
        batch_size=32, epochs=10, verbose=2)
        #menggunaakn batch ukuran 32, epochs 10 dan verbose 2
```

```
Train on 168233 samples
Epoch 1/10
168233/168233 - 69s - loss: 1.7824 - accuracy: 0.5859
Epoch 2/10
168233/168233 - 69s - loss: 1.0818 - accuracy: 0.7077
Epoch 3/10
168233/168233 - 71s - loss: 0.9625 - accuracy: 0.7305
Epoch 4/10
168233/168233 - 70s - loss: 0.9041 - accuracy: 0.7435
Epoch 5/10
168233/168233 - 69s - loss: 0.8627 - accuracy: 0.7521
Epoch 6/10
168233/168233 - 71s - loss: 0.8364 - accuracy: 0.7582
Epoch 7/10
168233/168233 - 70s - loss: 0.8129 - accuracy: 0.7635
Epoch 8/10
168233/168233 - 71s - loss: 0.7948 - accuracy: 0.7666
Epoch 9/10
168233/168233 - 70s - loss: 0.7816 - accuracy: 0.7687
Epoch 10/10
168233/168233 - 70s - loss: 0.7676 - accuracy: 0.7722
```

Gambar 7.104 Hasil Praktek 15

16. Jelaskan kode program pada blok In[16]

```
# In[16]:save the trained model
model.save("mathsymbols.model")
#Menyimpan model dengan nama mathsymbols.model
```

```
In [22]: runcell('[16]:save the trained model', 'D:/New folder/KB3C/src/1174084/7/1174084.py')
INFO:tensorflow:Assets written to: mathsymbols.model\assets
```

Gambar 7.105 Hasil Praktek 16

17. Jelaskan kode program pada blok In[17]

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

```
In [23]: runcell('[17]:save label encoder (to reverse one-hot encoding)', 'D:/New folder/KB3C/src/1174084/7/1174084.py')
```

Gambar 7.106 Hasil Praktek 17

18. Jelaskan kode program pada blok In[18]

```
# In[18]:load the pre-trained model and predict the math symbol for an arbitrary image;
# the code below could be placed in a separate file

import keras.models
#Mengimport library keras.models
model2 = keras.models.load_model("mathsymbols.model")
#Membuat variabel model2 dengan isi hasil load model dari mathsymbols.model
print(model2.summary())
#Menampilkan ringkasan dari model
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_24 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_23 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_25 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_24 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
flatten_19 (Flatten)         (None, 1152)              0
_________________________________________________________________
dense_36 (Dense)             (None, 128)               147584
_________________________________________________________________
dropout_13 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_37 (Dense)             (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.107 Hasil Praktek 18

19. Jelaskan kode program pada blok In[19]

```
# In[19]:restore the class name to integer encoder
label_encoder2 = LabelEncoder()
#Membuat variabel label_encoder ke 2 dengan isi fungsi Label_Encoder
label_encoder2.classes_ = np.load('classes.npy')
#Menggunakan method classess dengan data classess.npy yang di eksport.

def predict(img_path):
    #Membuat fungsi predict dengan path img
    newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path))
    #Membuat variable newping dengan isi mengubah bentuk image menjadi array dan membuka data berdasarkan img path
    newimg /= 255.0
    #Membagi data yang terdapat pada newimg dengan 255.0

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

```
#Membuat variabel prediction dengan isian variabel model2 menggunakan fungsi predict dengan syarat variabel newimg dengan data reshape

# 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
#Membuat variabel inverted  denagan label encoder2 dan menggunakan argmax untuk mencari skor luaran tertinggi
print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
#Menampilkan prediksi gambar dan confidence dari gambar.
```

```
In [30]: runcell('[19]:restore the class name to integer encoder', 'D:/New folder/KB3C/src/1174084/7/1174084.py')
```

Gambar 7.108 Hasil Praktek 19

20. Jelaskan kode program pada blok In[20]

```
# In[20]: grab an image (we'll just use a random training image for demonstration purposes)
predict("D:/New folder/KB3C/src/1174084/7/HASYv2/hasy-data/v2-00010.png")
#Melakukan prediksi dari pelatihan dari gambar v2-00010.png
predict("D:/New folder/KB3C/src/1174084/7/HASYv2/hasy-data/v2-00500.png")
#Melakukan prediksi dari pelatihan dari gambar v2-00500.png
predict("D:/New folder/KB3C/src/1174084/7/HASYv2/hasy-data/v2-00700.png")
#Melakukan prediksi dari pelatihan dari gambar v2-00700.png
```

```
In [36]: runcell("[20]: grab an image (we'll just use a random training image for demonstration purposes)", 'D:/New folder/KB3C/src/1174084/7/1174084.py')
Prediction: A, confidence: 0.48
Prediction: \pi, confidence: 0.54
Prediction: \alpha, confidence: 0.87
```

Gambar 7.109 Hasil Praktek 20

7.5.3 Bukti Tidak Plagiat

Gambar 7.110 plagiarism

7.5.4 Link Video Youtube

https://youtu.be/djRkJBfOFJs

7.6 1174070 - Arrizal Furqona Gifary

7.6.1 Soal Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
   Tokenizer untuk memisahkan input text menjadi potongan yang memiliki arti disebut tokenization. Hasil dari proses tokenization disebut token. Token dapat berupa kata, kalimat, paragraph dan lainnya. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.111 Teori 1

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

```

kfold = StratifiedKFold(n_splits=5)
```

   Konsep sederhana dari K Fold Cross Validation ialah Pada code tesebut terdapat kfold yang bertujuan untuk melakukan split data menjadi 5 bagian dari dataset komentar Youtube tersebut. Sehingga dari setiap data yang sudah dibagi tersebut akan menghasilkan presentase dari setiap bagiannya, untuk menghasilkan hasil akhir dengan presentase yang cukup baik. Untuk ilustrasi, lihat gambar berikut:

	A	B	C	D	E	F	G	H
1	DATA	HASIL					AKURASI DATA	
2	1						%	
3	2						%	
4	3						%	
5	4						%	
6	5						%	

Gambar 7.112 Teori 2

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

   Untuk penjelasan nya yaitu For train berfungsi untuk membagi data tersebut

menjadi data training. Sedangkan test in splits berfungsi untuk menguji apakah dataset tersebut sudah dibagi menjadi beberapa bagian atau masih menumpuk. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.113 Teori 3

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.

   Fungsi dalam kode tersebut berfungsi untuk mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Contoh sederhananya ketika data telah diubah menjadi data train dan data test maka kita dapat memilihnya untuk ditampilkan pada kolom yang di inginkan. Namun, untuk ilustrasi lihat gambar berikut:

```
import pandas as pd

mydict = [{'a': 1, 'b': 2, 'c': 3, 'd': 4},
          {'a': 100, 'b': 200, 'c': 300, 'd': 400},
          {'a': 1000, 'b': 2000, 'c': 3000, 'd': 4000 }]
df = pd.DataFrame(mydict)
df
```

```
Out[5]:
a    1
b    2
c    3
d    4
Name: 0, dtype: int64
```

Gambar 7.114 Teori 4

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 ini berfungsi untuk melakukan vektorisasi data kedalam bentuk token sebanyak 2000 kata. Dan selanjutnya akan melakukan fit tokenizer hanya untuk data training saja tidak dengan data testingnya. Untuk ilustrasi lihat gambar berikut:

Gambar 7.115 Teori 5

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.
   Maksud dari baris diatas ialah untuk memasukkan text ke sebuah matrix dengan mode tfidf dan menginputkan data testing untuk di terjemahkan ke sebuah matriks. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.116 Teori 6

7. Jelaskan apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train inputs)) dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs)), dilengkapi dengan ilustrasi atau gambar.
   Fungsi np.amax adalah nilai Maksimal. Jika sumbu tidak ada, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi a.ndim - 1. Untuk ilustrasi, lihat gambar berikut:

```
>>> a = np.arange(4).reshape((2,2))
>>> a
array([[0, 1],
       [2, 3]])
>>> np.amax(a)           # Maximum of the flattened array
3
>>> np.amax(a, axis=0)   # Maxima along the first axis
array([2, 3])
>>> np.amax(a, axis=1)   # Maxima along the second axis
array([1, 3])
>>> np.amax(a, where=[False, True], initial=-1, axis=0)
array([-1,  3])
>>> b = np.arange(5, dtype=float)
>>> b[2] = np.NaN
>>> np.amax(b)
nan
>>> np.amax(b, where=~np.isnan(b), initial=-1)
4.0
>>> np.nanmax(b)
4.0
```

Gambar 7.117 Teori 7

8. Jelaskan apa maksud fungsi dari d train outputs = np utils.to categorical(d[CLASS].iloc[tra idx]) dan d test outputs = np utils.to categorical(d[CLASS].iloc[test idx]) dalam kode program, dilengkapi dengan ilustrasi atau gambar.

Fungsi dari baris kode tersebut ialah membuat train outputs dengan kategori dari class lalu dengan ketentuan iloc train idx. Kemudian membuat keluaran sebagai output. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.118 Teori 8

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

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

Fungsi dari baris kode tersebut ialah model perlu mengetahui bentuk input apa yang harus diharapkan. Untuk alasan ini, lapisan pertama dalam model Sequential (dan hanya yang pertama, karena lapisan berikut dapat melakukan inferensi bentuk otomatis) perlu menerima informasi tentang bentuk inputnya.

Gambar 7.119 Teori 9

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

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

Fungsi dari baris kode tersebut ialah bisa meneruskan nama fungsi loss yang ada, atau melewati fungsi simbolis TensorFlow yang mengembalikan skalar untuk setiap titik data dan mengambil dua argumen y_true: True label. dan y_pred:

Prediksi. Tujuan yang dioptimalkan sebenarnya adalah rata-rata dari array output di semua titik data.

Gambar 7.120 Teori 10

11. Jelaskan apa itu Deep Learning.

    Deep Learning adalah salah satu cabang dari ilmu machine learning yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam machine learning dapat digunakan baik untuk supervised learning dan unsupervised learning.

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning.

    DNN adalah salah satu algoritma berbasis jaringan saraf tiruan yang memiliki dari 1 lapisan saraf tersembunyi yang dapat digunakan untuk pengambilan keputun. Perbedaannya dengan deep learning, yakni: DNN dapat menentukan dan mencerna karakteristik tertentu di suatu rangkaian data, kapabilitas lebih kompleks untuk mempelajari, mencerna, dan mengklasifikasikan data, serta dibagi ke dalam berbagai lapisan dengan fungsi yang berbeda-beda.

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.

    1174070 mod3+1 x 1174070 mod3+1 = 2 x 2, adapun ilustrasi gambar nya sebagai berikut :

Gambar 7.121 Teori 9

7.6.2 Praktek Program

1. Soal 1

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

Kode di atas menjelaskan tentang library yang akan di pakai yaitu import file csv, lalu load module pil image dan juga import library keras, hasilnya adalah sebagai berikut:

Gambar 7.122 Hasil Soal 1.

2. Soal 2

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

```
            #membagi data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # menambah nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # menambahkan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

Kode di atas akan menampilkan hasil dari proses load dataset dari HASYv2 sebagai file csv, membuat variabel i dengan parameter 0, lalu perintah perulangan if dengan i¿0, variabel img dengan nilai 255.0, imgs.append yaitu proses melampirkan atau menggabungkan data dengan file. Berikut adalah hasil setelah saya lakukan running dan pembacaan file audio :

3. Soal 3

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

Perintah import random yaitu sebagai library untuk menghasilkan nilai acak, disini kita membuat data menjadi 2 bagian yaitu 80% sebagai training dan 20% sebagai data testing. Hasilnya adalah sebagai berikut :

Gambar 7.123 Hasil Soal 3.

4. Soal 4

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

```
# membuat test input input dengan np method asarray yang mana
    membuat array dengan isi row 2 dari data test
test_input = np.asarray(list(map(lambda row: row[2], test)))
# membuat variabel train_output dengan np method asarray yang
    mana membuat array dengan isi row 1 dari data train
train_output = np.asarray(list(map(lambda row: row[1], train)))
# membuat variabel test_output dengan np method asarray yang mana
     membuat array dengan isi row 1 dari data test
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Kode di atas dapat digunakan untuk melakukan fungsi yang sebelumnya telah kita lakukan yaitu dengan membuat variabel baru untuk menampung data train input dan test input lalu keluaran nya sebagai output, . Hasilnya adalah sebagai berikut :

Gambar 7.124 Hasil Soal 4.

5. Soal 5

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

Kode diatas untuk melakukan import library yang berfungsi sebagai label encoder dan one hot encoder. Hasilnya adalah sebagai berikut :

Gambar 7.125 Hasil Soal 5.

6. Soal 6

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

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

Kode diatas berfungsi untuk melakukan convert class names ke one-hot encoding. Hasilnya adalah sebagai berikut :

```
In [47]: label_encoder = LabelEncoder()
   ...: # membuat variabel integer_encoded yang
berfungsi untuk mengkonvert variabel classes kedalam
bentuk integer
   ...: integer_encoded =
label_encoder.fit_transform(classes)
```

Gambar 7.126 Hasil Soal 6.

7. Soal 7

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

Kode di atas befungsi untuk melakukan convert integers ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

```
In [8]: onehot_encoder = OneHotEncoder(sparse=False)
   ...: integer_encoded = integer_encoded.reshape(len(integer_encoded), 1)
   ...: onehot_encoder.fit(integer_encoded)
Out[8]:
OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
              handle_unknown='error', sparse=False)
```

Gambar 7.127 Hasil Soal 7.

8. Soal 8

```
# In[8]:convert train and test output to one-hot
# mengkonvert data train output  mengguanakn variabel
    label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# mengkonvert variabel train_output_int kedalam fungsi
    onehot_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
# mengkonvert data test_output mengguanakn variabel label_encoder
     kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# mengkonvert variabel test_output_int kedalam fungsi
    onehot_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
# membuat variabel num_classes dengan isi variabel label_encoder
    dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

Kode di atas befungsi untuk melakukan convert train dan test output ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

Gambar 7.128 Hasil Soal 8.

9. Soal 9

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

Kode di atas befungsi untuk melakukan import sequential dari library keras. Hasilnya adalah sebagai berikut :

Gambar 7.129 Hasil Soal 9.

10. Soal 10

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

```
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

Kode di atas befungsi untuk melihat detail desain pada jaringan model yang telah di defenisikan. Hasilnya adalah sebagai berikut :

Gambar 7.130 Hasil Soal 10.

11. Soal 11

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

Kode di atas befungsi untuk melakukan load callbacks pada library keras. Hasilnya adalah sebagai berikut :

Gambar 7.131 Hasil Soal 11.

12. Soal 12

```
# In[12]: 5menit kali 10 epoch = 50 menit
# fungsi model titambahkan metod fit untuk mengetahui perhitungan dari train_input train_output
model.fit(train_input, train_output,
```

```
# dengan batch size 32 bit
            batch_size=32,
            epochs=10,
            verbose=2,
            validation_split=0.2,
            callbacks=[tensorboard])

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

Kode di atas befungsi untuk melakukan load epoch yang akan di training dan diberikan hasil selama 10 kali epoch. Hasilnya adalah sebagai berikut :

Gambar 7.132 Hasil Soal 12.

13. Soal 13

```
# In[13]:try various model configurations and parameters to find the best
# mengimport librari time
import time
#membuat variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # membuat variabel model Sequential
            model = Sequential()
            #membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
            for i in range(conv2d_count):
                # syarat jika i samadengan bobotnya 0
                if i == 0:
                    # menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
```

```
                    # jika tidak
                else:
                    # menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                # menambahkan method add pada variabel model dengan isian method  Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
                model.add(MaxPooling2D(pool_size=(2, 2)))
            # merubah feature gambar menjadi 1 dimensi vektor
            model.add(Flatten())
            # menambahkan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
            model.add(Dense(dense_size, activation='tanh'))
            # membuat ketentuan jika pemangkasan lebih besar dari 0 persen
            if dropout > 0.0:
                # menambahkan method dropout pada model dengan nilai dari dropout
                model.add(Dropout(dropout))
                # menambahkan method dense dengan fungsi num classs dan rumus softmax
            model.add(Dense(num_classes, activation='softmax'))
            # mongkompile variabel model dengan hasi loss optimasi dan akurasi matrix
            model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
            # melakukan log pada dir
            log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
            # membuat variabel tensorboard dengan isian dari librari keras dan nilai dari lig dir
            tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
            # membuat variabel start dengan isian dari librari time menggunakan method time

            start = time.time()
            # menambahkan method fit pada model dengan data dari train input train output nilai batch nilai epoch verbose nilai 20 persen validation split dan callback dengan nilai tnsorboard.
            model.fit(train_input, train_output, batch_size=32, epochs=10,
                      verbose=0, validation_split=0.2, callbacks=[tensorboard])
            # membuat variabel score dengan nilai evaluasi dari model menggunakan data tes input dan tes output
            score = model.evaluate(test_input, test_output, verbose=2)
            # membuat variabel end
            end = time.time()
            # membuat variabel elapsed
            elapsed = end - start
```

```
            # mencetak hasil perhitungan
            print("Conv2D count: %d, Dense size: %d, Dropout: %.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))
```

Kode di atas befungsi untuk melakukan konfigurasi model dengan parameter yang ditentukan dan mencoba mencari data yang terbaik. Hasilnya adalah sebagai berikut :

```
Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.16, Accuracy: 0.74, Time: 459 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.92, Accuracy: 0.70, Time: 447 sec
Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.82, Accuracy: 0.77, Time: 453 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: 1.38, Accuracy: 0.74, Time: 456 sec
Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.12, Accuracy: 0.70, 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: 1.68, Accuracy: 0.74, Time: 440 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.25 - Loss: 1.45, Accuracy: 0.75, Time: 466 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.50 - Loss: 1.24, Accuracy: 0.75, Time: 475 sec
Conv2D count: 1, Dense size: 512, Dropout: 0.75 - Loss: 0.89, Accuracy: 0.77, Time: 478 sec
Conv2D count: 1, Dense size: 1024, Dropout: 0.00 - Loss: 2.13, Accuracy: 0.72, Time: 498 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: 0.96, 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: 2.05, Accuracy: 0.70, Time: 629 sec
Conv2D count: 1, Dense size: 2048, Dropout: 0.50 - Loss: 1.55, Accuracy: 0.72, Time: 631 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: 0.70, 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: 0.76, Accuracy: 0.78, Time: 568 sec
Conv2D count: 2, Dense size: 128, Dropout: 0.75 - Loss: 0.76, Accuracy: 0.77, Time: 573 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.00 - Loss: 0.99, Accuracy: 0.70, Time: 550 sec
Conv2D count: 2, Dense size: 256, Dropout: 0.25 - Loss: 0.85, Accuracy: 0.77, Time: 581 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: 581 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: 1.05, 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: 618 sec
```

Gambar 7.133 Hasil Soal 13.

14. Soal 14

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

```
model.add(Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

Kode di atas befungsi untuk membuat data training kembali dengan model yang sudah di tentukan dan mencari yang terbaik dari hasil training tersebut. Hasilnya adalah sebagai berikut :

Gambar 7.134 Hasil Soal 14.

15. Soal 15

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

Kode di atas befungsi untuk melakukan join terhadap data train dan test dengan seluruh konfigurasi yang telah di tentukan sebelumnya. Hasilnya adalah sebagai berikut :

```
In [16]: model.fit(np.concatenate((train_input, test_input)),
    ...:                np.concatenate((train_output, test_output)),
    ...:                batch_size=4, epochs=10, verbose=1)
Epoch 1/10
 - 126s - loss: 1.7795 - accuracy: 0.5871
Epoch 2/10
 - 114s - loss: 1.0744 - accuracy: 0.7074
Epoch 3/10
 - 130s - loss: 0.9564 - accuracy: 0.7320
Epoch 4/10
 - 115s - loss: 0.8977 - accuracy: 0.7458
Epoch 5/10
 - 118s - loss: 0.8573 - accuracy: 0.7527
Epoch 6/10
 - 116s - loss: 0.8297 - accuracy: 0.7586
Epoch 7/10
 - 126s - loss: 0.8075 - accuracy: 0.7632
Epoch 8/10
 - 122s - loss: 0.7917 - accuracy: 0.7663
Epoch 9/10
 - 129s - loss: 0.7765 - accuracy: 0.7707
Epoch 10/10
 - 118s - loss: 0.7637 - accuracy: 0.7718
Out[16]: <keras.callbacks.callbacks.History at 0x21637ebc348>
```

Gambar 7.135 Hasil Soal 15.

16. Soal 16

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

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai mathsymbols.model. Hasilnya adalah sebagai berikut :

```
# save the trained model
model.save("mathsymbols.model")
```

Gambar 7.136 Hasil Soal 16.

17. Soal 17

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

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai classes.npy dengan reverse ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

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

Gambar 7.137 Hasil Soal 17.

18. Soal 18

```
# In[18]:load the pre-trained model and predict the math symbol
    for an arbitrary image;
# the code below could be placed in a separate file
# mengimpport librari keras model
import keras.models
```

```
# membuat variabel model2 untuk meload model yang telah di simpan tadi
model2 = keras.models.load_model("mathsymbols.model")
# mencetak hasil model2
print(model2.summary())
```

Kode di atas befungsi untuk melakukan load data yang sudah di train dan juga memprediksikan hasil. Hasilnya adalah sebagai berikut :

```
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
```

Gambar 7.138 Hasil Soal 18.

19. Soal 19

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

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

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

```
print("Prediction: %s, confidence: %.2f" % (inverted[0], np.max(prediction)))
```

Kode di atas befungsi untuk melakukan restore terhadap nama class pada fungsi integer encoder, dengan membuat fungsi predict. Hasilnya adalah sebagai berikut :

```
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
```

Gambar 7.139 Hasil Soal 19.

20. Soal 20

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

Kode di atas befungsi untuk menunjukkan hasil akhir prediski. Hasilnya adalah sebagai berikut :

```
# grab an image (we'll just use a random training image for demonstration purposes)
predict("HASYv2/hasy-data/v2-00010.png")
Prediction: A, confidence: 0.87
predict("HASYv2/hasy-data/v2-00500.png")
Prediction: \pi, confidence: 0.58
predict("HASYv2/hasy-data/v2-00700.png")
Prediction: \alpha, confidence: 0.88
```

Gambar 7.140 Hasil Soal 20.

7.6.3 Penanganan Error

1. KeyboardInterrupt

```
  File "<ipyn-input-20-2b0f0d0f0f0a>", line 19, in <module>
    img = keras.preprocessing.image.img_to_array(gil_image.open("HASYv2/" + row[0]))
  File "C:\Users\ASUS\ Pc\Anaconda3\lib\site-packages\PIL\Image.py", line 2809, in open
    fp = builtins.open(filename, "rb")
KeyboardInterrupt
```

Gambar 7.141 KeyboardInterrupt

2. Cara Penanganan Error

   - KeyboardInterrupt
     Error tersebut karena disebabkan oleh s yang melakukan klik pada keyboard saat running.

7.6.4 Bukti Tidak Plagiat

Gambar 7.142 Bukti Tidak Melakukan Plagiat Chapter 7

7.7 1174087 - Ilham Muhammad Ariq

7.7.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
   Untuk memudahkan mesin memahami maksud dari apa yang kita inginkan dalam machine learning, kata pada teks disebut token, dan proses vektorisasi dari bentuk kata ke dalam token tersebut disebut tokenizer dan tokenizer akan merubah sebuah teks menjadi simbol, kata, ataupun biner dan bentuk lainnya kedalam token. Ilutrasinya misalkan saya mempunyai sebuah kalimat yaitu Nama Saya Ilham Muhammad Ariq maka ketika kita lakukan proses tokenizer maka akan berubah menjadi [’Nama’, ’Saya’, ’Ilham’,’Muhammad’,’Ariq’].

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

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

```

Listing 7.1 K Fold Cross Validation

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

Gambar 7.143 Illustrasi K Fold Cross Validation

3. Jelaskan apa maksudnya kode program *for train, test in splits*.dilengkapi dengan ilustrasi atau gambar.
   Maksudnya yaitu untuk menguji apakah setiap data pada dataset sudah di split dan tidak terjadi penumpukan. Yang dimana maksudnya disetiap class tidak akan muncul id yang sama. Ilustrasinya misalkan kita memiliki 5 sepatu dengan model yang berbeda. Kemudian kita bagikan ke kedua orang , tentunya setiap orang yang menerima sepatu tidak memiliki model sepatu yang sama.

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.
   Maksudnya yaitu mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train idx dan test idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memilihnya untuk ditampilkan pada kolom yang diinginkan.

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

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

Gambar 7.144 Illustrasi fit tokenizer dan num_word=2000

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

Gambar 7.145 Illustrasi d train inputs = tokenizer.texts to matrix

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

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

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

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

```

Listing 7.2 Membuat model Neural Network

Penjelasannya sebagai berikut:

- Melakukan pemodelan Sequential
- Layer pertama dense dari 512 neuron untuk inputan dengan inputan tadi yang sudah dijadikan matriks sebanyak 2000
- Activationnya menggunakan fungsi relu yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.
- Dropout ini untuk melakukan pembobotan,dimana pembobotan hanya dilakukan 50 persen saja agar tidak terjadi penumpukan data dari dense inputan tadi

- Dense 2 mengkategorikan 2 neuron untuk output nya yaitu 1 dan 0.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax.

Gambar 7.146 Illustrasi Neural Network Pemodelan

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

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

```

Listing 7.3 Compile model

Melakukan peng compile-an dari model Sequential tadi dengan Loss yandengang merupakanfungsioptimisasiskormenggunakancategorical crossentropy,danmenggunaalgoritma adam sebagai optimizer. Adam yaitu algoritma pengoptimalan yangdapatdigunakansebagaigantidariprosedurpenurunangradienstokastikklasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training.Dengan metrik yaitu fungsi yang digunakan untuk menilai kinerja mode Anda disini menggunakan fungsi accuracy.

11. Jelaskan apa itu Deep Learning
    Deep Learning adalah subbidang machine learning yang berkaitan dengan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan atau Articial Neural Networks. Jaringan saraf tiruan, algoritma yang terinspirasi oleh otak manusia, belajar dari sejumlah besar data. Demikian pula dengan bagaimana kita belajar dari pengalaman, algoritma pembelajaran yang mendalam akan melakukan tugas berulang kali,setiap kali sedikit mengubahnya untuk meningkatkan hasilnya.

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning
    Deep Neural Network adalah jaringan syaraf tiruan (JST) dengan beberapa lapisan antara lapisan input dan output. DNN menemukan manipulasi matematis yang benar untuk mengubah input menjadi output, apakah itu hubungan lin-

ear atau hubungan non-linear. Merupakan jaringan syaraf dengan tingkat kompleksitas tertentu, jaringan syaraf dengan lebih dari dua lapisan. Deep Neural Network menggunakan pemodelan matematika yang canggih untuk memproses data dengan cara yang kompleks.
DNN hanya terdiri dari dua laipsan yaitu input dan output, sedangkan dalam Deep learning kita dapat mendeniskan layer sebanyak yang kita inginkan atau butuhkan.

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

Gambar 7.147 Illustrasi perhitungan stride 1 max pooling

7.7.2 Praktek

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

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

```
import keras.preprocessing.image
```

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

```
# In[2]:load all images (as numpy arrays) and save their classes
#Menginisiasi variabel imgs dan classes dengan variabel array kosong
imgs = []
classes = []
#membuaka file hasy-data-labels.csv yang berada di folede HASYv2 yang di inisialisasi menjadi csvfile
with open('HASYv2/hasy-data-labels.csv') as csvfile:
    #Menginisiasi variabel csvreader yang berisi method csv.reader yang membaca variabel csvfile
    csvreader = csv.reader(csvfile)
    # Menginisiasi variabel i dengan isi 0
    i = 0
    # membuat looping pada variabel csvreader
    for row in csvreader:
        # dengan ketentuan jika i lebihkecil daripada o
        if i > 0:
            # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
            #Pembagian data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # Penambahan nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # Penambahan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

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

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

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

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

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

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

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

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

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

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

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

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

```
# In[8]:convert train and test output to one-hot
# Menconvert data train output  mengguanakn variabel label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# Menconvert variabel train_output_int kedalam fungsi onehot_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1))
# Menconvert data test_output mengguanakn variabel label_encoder kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# Menconvert variabel test_output_int kedalam fungsi onehot_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1))
# Menginisiasi variabel num_classes dengan isi variabel label_encoder dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

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

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

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

```
# In[10]: desain jaringan
# Menginisiasi variabel model dengan isian library Sequential
model = tf.keras.Sequential()
# variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
```

```
# variabel model di tambahkan dengan library Conv2D 32bit dengan kernel 3 x 3
model.add(tf.keras.layers.Conv2D(32, (3, 3), activation='relu'))
# variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
# variabel model di tambahkan library Flatten
model.add(tf.keras.layers.Flatten())
# variabel model di tambahkan library Dense dengan fungsi tanh
model.add(tf.keras.layers.Dense(1024, activation='tanh'))
# variabel model di tambahkan library dropout untuk memangkas data tree sebesar 50 persen
model.add(tf.keras.layers.Dropout(0.5))
# variabel model di tambahkan library Dense dengan data dari num_classes dan fungsi softmax
model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

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

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

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

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

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

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

```
# In[13]:try various model configurations and parameters to find the best
# Melakukan import library time
import time
#Menginisiasi variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # Menginisiasi variabel model Sequential
            model = tf.keras.Sequential()
            #membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
            for i in range(conv2d_count):
                # syarat jika i samadengan bobotnya 0
                if i == 0:
                    # Penambahan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                    model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
                    # jika tidak
                else:
                    # Penambahan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                    model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu'))
                # Penambahan method add pada variabel model dengan isian method  Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
                model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
            # merubah feature gambar menjadi 1 dimensi vektor
            model.add(tf.keras.layers.Flatten())
            # Penambahan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
            model.add(tf.keras.layers.Dense(dense_size, activation='tanh'))
            # membuat ketentuan jika pemangkasan lebih besar dari 0 persen
            if dropout > 0.0:
                # Penambahan method dropout pada model dengan nilai dari dropout
                model.add(tf.keras.layers.Dropout(dropout))
```

```
                # Penambahan method dense dengan fungsi num classs dan rumus softmax
            model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
            # mongkompile variabel model dengan hasi loss optimasi dan akurasi matrix
            model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
            # melakukan log pada dir
            log_dir = '.\\logs\\conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
            # Menginisiasi variabel tensorboard dengan isian dari library keras dan nilai dari lig dir
            tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
            # Menginisiasi variabel start dengan isian dari library time menggunakan method time

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

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

```
# In[14]:rebuild/retrain a model with the best parameters (from the search) and use all data
# Menginisiasi variabel model dengan isian library Sequential
model = tf.keras.Sequential()
# variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.7.3 Penanganan Error

1. SS Error

Gambar 7.148 File Not Found error

2. Jenis Error
   - File Not Found Error
3. Cara Penanganan
   Dengan cara menyesuaikan letak file yang ingin dibaca/ digunakan

7.7.4 Bukti Tidak Plagiat

Gambar 7.149 Tidak Melakukan Plagiat Pada Ch 7

7.7.5 Link Video Youtube

https://youtu.be/XPdKuIEMeu8

7.8 1174073 - Ainul Filiani

7.8.1 Soal Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
   Karena MTokenizer merupakan proses membagi teks yang dapat berupa kalimat, paragraf atau dokumen menjadi kata-kata atau bagian-bagian tertentu dalam kalimat tersebut. Sebagai contoh dari kalimat ”Cieee yang copas punya ainul”, kalimat itu menjadi beberapa bagian yaitu ”cieee”, ”yang”, ”copas”, ”punya”, ”ainul”. Yang menjadi acuan yakni tanda baca dan spasi. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.150 Teori 1

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

```
model.add(Dense(2))
model.add(Activation('softmax'))
```

   Konsep sederhana dari K Fold Cross Validation ialah Pada code tesebut terdapat kfold yang bertujuan untuk melakukan split data menjadi 5 bagian dari dataset komentar Youtube tersebut. Sehingga dari setiap data yang sudah dibagi tersebut akan menghasilkan presentase dari setiap bagiannya, untuk menghasilkan hasil akhir dengan presentase yang cukup baik. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.151 Teori 2

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

Untuk penjelasan nya yaitu For train berfungsi untuk membagi data tersebut menjadi data training. Sedangkan test in splits berfungsi untuk menguji apakah dataset tersebut sudah dibagi menjadi beberapa bagian atau masih menumpuk. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.152 Teori 3

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.

   Fungsi dalam kode tersebut berfungsi untuk mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Contoh sederhananya ketika data telah diubah menjadi data train dan data test maka kita dapat memilihnya untuk ditampilkan pada kolom yang di inginkan. Namun, untuk ilustrasi lihat gambar berikut:

Gambar 7.153 Teori 4

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 ini berfungsi untuk melakukan vektorisasi data kedalam bentuk token sebanyak 2000 kata. Dan selanjutnya akan melakukan fit tokenizer hanya untuk data training saja tidak dengan data testingnya. Untuk ilustrasi lihat gambar berikut:

Gambar 7.154 Teori 5

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.
   Maksud dari baris diatas ialah untuk memasukkan text ke sebuah matrix dengan mode tfidf dan menginputkan data testing untuk di terjemahkan ke sebuah matriks. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.155 Teori 6

7. Jelaskan apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train inputs)) dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs)), dilengkapi dengan ilustrasi atau gambar.
   Fungsi np.amax adalah nilai Maksimal. Jika sumbu tidak ada, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi a.ndim - 1. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.156 Teori 7

8. Jelaskan apa maksud fungsi dari d train outputs = np utils.to categorical(d[CLASS].iloc[tra idx]) dan d test outputs = np utils.to categorical(d[CLASS].iloc[test idx]) dalam kode program, dilengkapi dengan ilustrasi atau gambar.

Fungsi dari baris kode tersebut ialah membuat train outputs dengan kategori dari class lalu dengan ketentuan iloc train idx. Kemudian membuat keluaran sebagai output. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.157 Teori 8

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

Fungsi dari baris kode tersebut ialah model perlu mengetahui bentuk input apa yang harus diharapkan. Untuk alasan ini, lapisan pertama dalam model Sequential (dan hanya yang pertama, karena lapisan berikut dapat melakukan inferensi bentuk otomatis) perlu menerima informasi tentang bentuk inputnya.

Gambar 7.158 Teori 9

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

Fungsi dari baris kode tersebut ialah bisa meneruskan nama fungsi loss yang ada, atau melewati fungsi simbolis TensorFlow yang mengembalikan skalar untuk setiap titik data dan mengambil dua argumen y_true: True label. dan y_pred: Prediksi. Tujuan yang dioptimalkan sebenarnya adalah rata-rata dari array output di semua titik data.

Gambar 7.159 Teori 10

11. Jelaskan apa itu Deep Learning.

    Deep Learning adalah salah satu cabang dari ilmu machine learning yang terdiri dari algoritma pemodelan abstraksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi non-linear yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam machine learning dapat digunakan baik untuk supervised learning dan unsupervised learning.

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning.

    DNN adalah salah satu algoritma berbasis jaringan saraf tiruan yang memiliki dari 1 lapisan saraf tersembunyi yang dapat digunakan untuk pengambilan keputun. Perbedaannya dengan deep learning, yakni: DNN dapat menentukan dan mencerna karakteristik tertentu di suatu rangkaian data, kapabilitas lebih kompleks untuk mempelajari, mencerna, dan mengklasifikasikan data, serta dibagi ke dalam berbagai lapisan dengan fungsi yang berbeda-beda.

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.

    1174073 mod3+1 x 1174073 mod3+1 = 2 x 2, adapun ilustrasi gambar nya sebagai berikut :

Gambar 7.160 Teori 9

7.8.2 Praktek Program

1. Soal 1

```
# In[1]:import lib
import keras.models
import time
import keras.callbacks
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Dense, Dropout, Flatten
from keras.models import Sequential
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
import numpy as np
import random
import csv
import tensorflow as tf
from PIL import Image as pil_image
import keras.preprocessing.image
```

Kode di atas menjelaskan tentang library yang akan di pakai yaitu import file csv, lalu load module pil image dan juga import library keras, hasilnya adalah sebagai berikut:

Gambar 7.161 Hasil Soal 1.

2. Soal 2

```
# In[2]:load all images (as numpy arrays) and save their classes

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
```

Kode di atas akan menampilkan hasil dari proses load dataset dari HASYv2 sebagai file csv, membuat variabel i dengan parameter 0, lalu perintah perulangan if dengan i¿0, variabel img dengan nilai 255.0, imgs.append yaitu proses melampirkan atau menggabungkan data dengan file. Berikut adalah hasil setelah saya lakukan running dan pembacaan file audio :

Gambar 7.162 Hasil Soal 2.

3. Soal 3

```
# In[3]:shuffle the data, split into 80% train, 20% test

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

Perintah import random yaitu sebagai library untuk menghasilkan nilai acak, disini kita membuat data menjadi 2 bagian yaitu 80% sebagai training dan 20% sebagai data testing. Hasilnya adalah sebagai berikut :

Gambar 7.163 Hasil Soal 3.

4. Soal 4

```
# In[4]:


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)))
```

Kode di atas dapat digunakan untuk melakukan fungsi yang sebelumnya telah kita lakukan yaitu dengan membuat variabel baru untuk menampung data train input dan test input lalu keluaran nya sebagai output, . Hasilnya adalah sebagai berikut :

Gambar 7.164 Hasil Soal 4.

5. Soal 5

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

Kode diatas untuk melakukan import library yang berfungsi sebagai label encoder dan one hot encoder. Hasilnya adalah sebagai berikut :

```
In [6]: from sklearn.preprocessing import LabelEncoder
   ...: from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.165 Hasil Soal 5.

6. Soal 6

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

# first, convert class names into integers
label_encoder = LabelEncoder()
integer_encoded = label_encoder.fit_transform(classes)
```

Kode diatas berfungsi untuk melakukan convert class names ke one-hot encoding. Hasilnya adalah sebagai berikut :

```
In [47]: label_encoder = LabelEncoder()
    ...: # membuat variabel integer_encoded yang
berfungsi untuk mengkonvert variabel classes kedalam
bentuk integer
    ...: integer_encoded =
label_encoder.fit_transform(classes)
```

Gambar 7.166 Hasil Soal 6.

7. Soal 7

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

Kode di atas befungsi untuk melakukan convert integers ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

```
In [8]: onehot_encoder = OneHotEncoder(sparse=False)
    ...: integer_encoded = integer_encoded.reshape(-1, (integer_encoded), 1)
    ...: onehot_encoder.fit(integer_encoded)
Out[8]:
OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
              handle_unknown='error', sparse=False)
```

Gambar 7.167 Hasil Soal 7.

8. Soal 8

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

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

Kode di atas befungsi untuk melakukan convert train dan test output ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

Gambar 7.168 Hasil Soal 8.

9. Soal 9

```
# In[9]: import sequential
```

Kode di atas befungsi untuk melakukan import sequential dari library keras. Hasilnya adalah sebagai berikut :

Gambar 7.169 Hasil Soal 9.

10. Soal 10

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

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

```

print(model.summary())
```

Kode di atas befungsi untuk melihat detail desain pada jaringan model yang telah di defenisikan. Hasilnya adalah sebagai berikut :

Gambar 7.170 Hasil Soal 10.

11. Soal 11

```
# In[11]: import sequential

tensorboard = keras.callbacks.TensorBoard(log_dir='.\\logs\\mnist
    -style')
```

Kode di atas befungsi untuk melakukan load callbacks pada library keras. Hasilnya adalah sebagai berikut :

Gambar 7.171 Hasil Soal 11.

12. Soal 12

```
# In[12]: 5menit kali 10 epoch = 50 menit
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])
```

Kode di atas befungsi untuk melakukan load epoch yang akan di training dan diberikan hasil selama 10 kali epoch. Hasilnya adalah sebagai berikut :

```
Epoch 1/10
 - 177s - loss: 1.5016 - accuracy: 0.6203 - val_loss: 1.0332 - val_accuracy: 0.7182
WARNING:tensorflow:From C:\Users\ Aqil \Anaconda3\lib\site-packages\keras\callbacks
\tensorboard_v1.py:343: The name tf.summary is deprecated. Please use tf.compat.v1.summary
instead.

Epoch 2/10
 - 179s - loss: 1.0001 - accuracy: 0.7348 - val_loss: 0.9099 - val_accuracy: 0.7523
Epoch 3/10
 - 180s - loss: 0.8901 - accuracy: 0.7677 - val_loss: 0.8052 - val_accuracy: 0.7562
Epoch 4/10
 - 212s - loss: 0.8220 - accuracy: 0.7612 - val_loss: 0.8360 - val_accuracy: 0.7629
Epoch 5/10
 - 192s - loss: 0.7751 - accuracy: 0.7716 - val_loss: 0.8479 - val_accuracy: 0.7638
Epoch 6/10
 - 177s - loss: 0.7316 - accuracy: 0.7804 - val_loss: 0.8502 - val_accuracy: 0.7642
Epoch 7/10
 - 189s - loss: 0.7034 - accuracy: 0.7859 - val_loss: 0.8566 - val_accuracy: 0.7654
Epoch 8/10
 - 198s - loss: 0.6765 - accuracy: 0.7924 - val_loss: 0.8492 - val_accuracy: 0.7639
Epoch 9/10
 - 177s - loss: 0.6512 - accuracy: 0.7985 - val_loss: 0.8531 - val_accuracy: 0.7681
Epoch 10/10
 - 175s - loss: 0.6312 - accuracy: 0.8021 - val_loss: 0.8792 - val_accuracy: 0.7611
Test loss: 0.8820041881901263
Test accuracy: 0.76054320680023445
```

Gambar 7.172 Hasil Soal 12.

13. Soal 13

```
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(128, activation='tanh'))
model.add(tf.keras.layers.Dropout(0.5))
model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer='adam', metrics=['accuracy'])
print(model.summary())
# In[15]:join train and test data so we train the network on all data we have available to us
model.fit(np.concatenate((train_input, test_input)),
          np.concatenate((train_output, test_output)),
          batch_size=32, epochs=10, verbose=2)

# In[16]:save the trained model
model.save("mathsymbols.model")

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


# In[18]:load the pre-trained model and predict the math symbol for an arbitrary image;
# the code below could be placed in a separate file

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

# In[19]:restore the class name to integer encoder
label_encoder2 = LabelEncoder()
label_encoder2.classes_ = np.load('classes.npy')


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

```

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

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


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

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

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

Kode di atas befungsi untuk melakukan konfigurasi model dengan parameter yang ditentukan dan mencoba mencari data yang terbaik. Hasilnya adalah sebagai berikut :

Gambar 7.173 Hasil Soal 13.

14. Soal 14

```
# In[14]:rebuild/retrain a model with the best parameters (from the search) and use all data
model = tf.keras.Sequential()
model.add(tf.keras.layers.Conv2D(32, kernel_size=(3, 3), activation='relu',
                 input_shape=np.shape(train_input[0])))
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Conv2D(32, (3, 3), activation='relu'))
model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(128, activation='tanh'))
model.add(tf.keras.layers.Dropout(0.5))
model.add(tf.keras.layers.Dense(num_classes, activation='softmax'))
```

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

Kode di atas befungsi untuk membuat data training kembali dengan model yang sudah di tentukan dan mencari yang terbaik dari hasil training tersebut. Hasilnya adalah sebagai berikut :

Gambar 7.174 Hasil Soal 14.

15. Soal 15

```
# In[15]:join train and test data so we train the network on all
    data we have available to us
model.fit(np.concatenate((train_input, test_input)),
          np.concatenate((train_output, test_output)),
          batch_size=32, epochs=10, verbose=2)
```

Kode di atas befungsi untuk melakukan join terhadap data train dan test dengan seluruh konfigurasi yang telah di tentukan sebelumnya. Hasilnya adalah sebagai berikut :

Gambar 7.175 Hasil Soal 15.

16. Soal 16

```
# In[16]:save the trained model
model.save("mathsymbols.model")
```

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai mathsymbols.model. Hasilnya adalah sebagai berikut :

```
# save the trained model
model.save("mathsymbols.model")
```

Gambar 7.176 Hasil Soal 16.

17. Soal 17

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

Kode di atas befungsi untuk melakukan save pada model yang akan disimpan sebagai classes.npy dengan reverse ke fungsi one hot encoding. Hasilnya adalah sebagai berikut :

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

Gambar 7.177 Hasil Soal 17.

18. Soal 18

```
# In[18]:load the pre-trained model and predict the math symbol
    for an arbitrary image;
# the code below could be placed in a separate file

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

Kode di atas befungsi untuk melakukan load data yang sudah di train dan juga memprediksikan hasil. Hasilnya adalah sebagai berikut :

```
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
```

Gambar 7.178 Hasil Soal 18.

19. Soal 19

```
# In[19]: restore the class name to integer encoder
label_encoder2 = LabelEncoder()
label_encoder2.classes_ = np.load('classes.npy')


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

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

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

Kode di atas befungsi untuk melakukan restore terhadap nama class pada fungsi integer encoder, dengan membuat fungsi predict. Hasilnya adalah sebagai berikut :

```
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
```

Gambar 7.179 Hasil Soal 19.

20. Soal 20

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

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

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

Kode di atas befungsi untuk menunjukkan hasil akhir prediski. Hasilnya adalah sebagai berikut :

Gambar 7.180 Hasil Soal 20.

7.8.3 Penanganan Error

1. KeyboardInterrupt

Gambar 7.181 KeyboardInterrupt

2. Cara Penanganan Error

   - KeyboardInterrupt
     Error tersebut karena disebabkan oleh human typo yang melakukan klik pada keyboard saat running.

7.8.4 Bukti Tidak Plagiat

Gambar 7.182 Bukti Tidak Melakukan Plagiat Chapter 7

7.9 1174077 - Alvan Alvanzah

7.9.1 Soal Teori

1. Jelaskan kenapa le teks harus di lakukan tokenizer
   Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
   Tokenizer diperlukan untuk perhitungan bobot pada setiap teks karena Tokenizer akan membagi kalimat menjadi beberapa teks sehingga nantinya kalimat

yang dimasukkan otomatis langsung diubah menjadi kata - kata dan akan dinilai sehingga akan memunculkan nilai vektor untuk digunakan saat prediksi teks yang muncul pada satu kalimat tersebut. Biasanya untuk memisahkan sebuah kata, tokenizer akan menggunakan spasi sebagai pemisah antar kata.

Gambar 7.183 Ilustrasi Tokenizer

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1 **??**

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

```

Listing 7.4 K Fold Cross Validation

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

Gambar 7.184 ilustrasi K-Fold Cross Validation

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

```
In [3]: import numpy as np
   ...: from sklearn.model_selection import train_test_split
   ...: X= np.arange(6).reshape((3, 2))
   ...: y = range(3)
   ...: print(X)
[[0 1]
 [2 3]
 [4 5]]

In [4]: list(y)
   ...: x_train, x_test, y_train, y_test = train_test_split(X,y,test_size=0.33, random_state=42)
   ...: print(x_train)
   ...: print(x_test)
[[2 3]
 [4 5]]
[[0 1]]
```

Gambar 7.185 Ilustrasi for train dan test in split

4. Jelaskan apa maksudnya kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx].
   Maksud dari kode program tersebut adalah membaca isian kolom pada eld yang bernama CONTENT sebagai data training dan data testing untuk program

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

Gambar 7.186 ilustrasi penggunaan kolom content

5. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.t on texts(train content).

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

Gambar 7.187 Ilustrasi Fit Tokenizer dan num_word=2000

6. Jelaskan apa maksud dari fungsi d train inputs = tokenizer.texts to matrix(train content, mode=tdf) dan d test inputs = tokenizer.texts to matrix(test content, mode=tdf),
   Untuk digunakan sebagai pengubah urutan teks yang tadi telah dilakukan tkoenizer menjadi matriks yang berurutan seperti tf idf

Gambar 7.188 Ilustrasi d train inputs = tokenizer.texts to matrix

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

Gambar 7.189 fungsi d train inputs

8. Jelaskan apa maksud fungsi dari d train outputs = np utils.to categorical(d[CLASS].iloc[tra dan d test outputs = np utils.to categorical(d[CLASS].iloc[test idx]) dalam kode program
   Maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing.

Gambar 7.190 fungsi train outputs = np utils.to categorical

9. Jelaskan apa maksud dari fungsi di listing 7.2 **??**.

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

Listing 7.5 Membuat model Neural Network

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

Gambar 7.191 ilustrasi neural network

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

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

```

Listing 7.6 Compile model

model tersebut kemudian di compile atau di kembalikan kembali fungsi nilainya yangmana akan mengembalikan fungsi nilai loss nya berapa yang diambil dari fungsi adamax yang berberguna untuk mengetahui nilai lossnya kemudian metrics = acuracy merupakan akurasi dari nilai matrixnya. kemudian terdiri atas beberapa layer atau hiden layer. perbedaan antara deep learning dan DNN atau Deep Neural Network yaitu deep lerning merupakan pemakai algoritma dari DNN dan DNN merupakan algoritma yang ada pada deep learning.

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

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

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

Gambar 7.192 ilustrasi perhitungan stride 1 max pooling

7.9.2 Praktek Program

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

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

Gambar 7.193 Hasil Soal 1.

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

```
# In[2]:load all images (as numpy arrays) and save their classes
#Menginisiasi variabel imgs dan classes dengan variabel array
    kosong
imgs = []
classes = []
```

```
#membuaka file hasy-data-labels.csv yang berada di folede HASYv2 yang di inisialisasi menjadi csvfile
with open('HASYv2/hasy-data-labels.csv') as csvfile:
    #Menginisiasi variabel csvreader yang berisi method csv.reader yang membaca variabel csvfile
    csvreader = csv.reader(csvfile)
    # Menginisiasi variabel i dengan isi 0
    i = 0
    # membuat looping pada variabel csvreader
    for row in csvreader:
        # dengan ketentuan jika i lebihkecil daripada o
        if i > 0:
            # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
            #Pembagian data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # Penambahan nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # Penambahan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1

# In[3]:shuffle the data, split into 80% train, 20% test
# Melakukan import library random
import random
```

```
In [34]: imgs = []
   ...: classes = []
   ...: #membuaka file hasy-data-labels.csv yang berada di folede HASYv2 yang di inisialisasi menjadi csvfile
   ...: with open('hasy-data-labels.csv') as csvfile:
   ...:     #Menginisiasi variabel csvreader yang berisi method csv.reader yang membaca variabel csvfile
   ...:     csvreader = csv.reader(csvfile)
   ...:     # Menginisiasi variabel i dengan isi 0
   ...:     i = 0
   ...:     # membuat looping pada variabel csvreader
   ...:     for row in csvreader:
   ...:         # dengan ketentuan jika i lebihkecil daripada o
   ...:         if i > 0:
   ...:             # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
   ...:             img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
   ...:             #Pembagian data yang ada pada fungsi img sebanyak 255.0
   ...:             img /= 255.0
   ...:             # Penambahan nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
   ...:             imgs.append((row[0], row[2], img))
   ...:             # Penambahan nilai pada row ke 2 pada variabel classes
   ...:             classes.append(row[2])
   ...:             # penambahan nilai satu pada variabel i
   ...:         i += 1
```

Gambar 7.194 Hasil Soal 2.

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.

```
# melakukan random pada vungsi imgs
random.shuffle(imgs)
```

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

# In[4]:
# Melakukan import library numpy dengan inisial np
import numpy as np
```

```
In [2]: import random
   ...: # melakukan random pada vungsi imgs
   ...: random.shuffle(imgs)
   ...: # Menginisiasi variabel split_idx dengan nilai integer 80 persen
dikali dari pengembalian jumlah dari variabel imgs
   ...: split_idx = int(0.8*len(imgs))
   ...: # Menginisiasi variabel train dengan isi lebih besar split idx
   ...: train = imgs[:split_idx]
   ...: # Menginisiasi variabel test dengan isi lebih kecil split idx
   ...: test = imgs[split_idx:]
```

Gambar 7.195 Hasil Soal 3.

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.

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

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

```
In [8]: import numpy as np
   ...: # Menginisiasi variabel train input dengan np method asarray yang
mana membuat array dengan isi row 2 dari data train
   ...: train_input = np.asarray(list(map(lambda row: row[2], train)))
   ...: # membuat test input input dengan np method asarray yang mana
membuat array dengan isi row 2 dari data test
   ...: test_input = np.asarray(list(map(lambda row: row[2], test)))
   ...: # Menginisiasi variabel train_output dengan np method asarray
yang mana membuat array dengan isi row 1 dari data train
   ...: train_output = np.asarray(list(map(lambda row: row[1], train)))
   ...: # Menginisiasi variabel test_output dengan np method asarray yang
mana membuat array dengan isi row 1 dari data test
   ...: test_output = np.asarray(list(map(lambda row: row[1], test)))
```

Gambar 7.196 Hasil Soal 4.

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 kom-

puter sendiri.

```
# Melakukan import library OneHotEncoder dari sklearn
from sklearn.preprocessing import OneHotEncoder

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

# Menginisiasi variabel label_encoder dengan isi LabelEncoder
```

```
In [9]: from sklearn.preprocessing import LabelEncoder
   ...: # Melakukan import library OneHotEncoder dari sklearn
   ...: from sklearn.preprocessing import OneHotEncoder
```

Gambar 7.197 Hasil Soal 5.

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.

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

# In[7]:then convert integers into one-hot encoding
# Menginisiasi variabel onehot_encoder dengan isi OneHotEncoder
onehot_encoder = OneHotEncoder(sparse=False)
```

```
In [10]: label_encoder = LabelEncoder()
   ...: # Menginisiasi variabel integer_encoded yang berfungsi untuk
Menconvert variabel classes kedalam bentuk integer
   ...: integer_encoded = label_encoder.fit_transform(classes)
```

Gambar 7.198 Hasil Soal 6.

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

```

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

```
In [11]: onehot_encoder = OneHotEncoder(sparse=False)
    ...: # mengisi variabel integer_encoded dengan isi integer_encoded
yang telah di convert pada fungsi sebelumnya
    ...: integer_encoded = integer_encoded.reshape(len(integer_encoded),
1)
    ...: # Menconvert variabel integer_encoded kedalam onehot_encoder
    ...: onehot_encoder.fit(integer_encoded)
D:\Software\Anaconda\lib\site-packages\sklearn\preprocessing
\_encoders.py:415: FutureWarning: The handling of integer data will
change in version 0.22. Currently, the categories are determined based o
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[11]:
OneHotEncoder(categorical_features=None, categories=None, drop=None,
              dtype=<class 'numpy.float64'>, handle_unknown='error',
              n_values=None, sparse=False)
```

Gambar 7.199 Hasil Soal 7.

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

```
# In[8]:convert train and test output to one-hot
# Menconvert data train output  mengguanakn variabel
    label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# Menconvert variabel train_output_int kedalam fungsi
    onehot_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
# Menconvert data test_output mengguanakn variabel label_encoder
    kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# Menconvert variabel test_output_int kedalam fungsi
    onehot_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
# Menginisiasi variabel num_classes dengan isi variabel
    label_encoder dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

```
In [12]: train_output_int = label_encoder.transform(train_output)
    ...: # Menconvert variabel train_output_int kedalam fungsi
onehot_encoder
    ...: train_output =
onehot_encoder.transform(train_output_int.reshape(len(train_output_int),
1))
    ...: # Menconvert data test_output mengguanakn variabel label_encoder
kedalam variabel test_output_int
    ...: test_output_int = label_encoder.transform(test_output)
    ...: # Menconvert variabel test_output_int kedalam fungsi
onehot_encoder
    ...: test_output =
onehot_encoder.transform(test_output_int.reshape(len(test_output_int),
1))
    ...: # Menginisiasi variabel num_classes dengan isi variabel
label_encoder dan classess
    ...: num_classes = len(label_encoder.classes_)
    ...: # mencetak hasil dari nomer Class beruapa persen
    ...: print("Number of classes: %d" % num_classes)
Number of classes: 369
```

Gambar 7.200 Hasil Soal 8.

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

```
# In[9]: import sequential
```

```
# Melakukan import library Sequential dari Keras
from keras.models import Sequential
# Melakukan import library Dense, Dropout, Flatten dari Keras
from keras.layers import Dense, Dropout, Flatten
# Melakukan import library Conv2D, MaxPooling2D dari Keras
from keras.layers import Conv2D, MaxPooling2D
```

```
In [13]: from keras.models import Sequential
    ...: # Melakukan import library Dense, Dropout, Flatten dari Keras
    ...: from keras.layers import Dense, Dropout, Flatten
    ...: # Melakukan import library Conv2D, MaxPooling2D dari Keras
    ...: from keras.layers import Conv2D, MaxPooling2D
```

Gambar 7.201 Hasil Soal 9.

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

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

```
...: print(model.summary())
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (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 7.202 Hasil Soal 10.

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

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

```
In [15]: import keras.callbacks
    ...: # Menginisiasi variabel tensorboard dengan isi lib keras
    ...: tensorboard = keras.callbacks.TensorBoard(log_dir='./logs/mnist-
style')
```

Gambar 7.203 Hasil Soal 11.

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

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

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

```
107668/107668 - 88s - loss: 1.5529 - accuracy: 0.6278 - val_loss: 0.9929
- val_accuracy: 0.7288
Epoch 2/10
107668/107668 - 81s - loss: 0.9793 - accuracy: 0.7307 - val_loss: 0.8903
- val_accuracy: 0.7518
Epoch 3/10
107668/107668 - 78s - loss: 0.8683 - accuracy: 0.7533 - val_loss: 0.8577
- val_accuracy: 0.7545
Epoch 4/10
107668/107668 - 82s - loss: 0.7983 - accuracy: 0.7676 - val_loss: 0.8427
- val_accuracy: 0.7629
Epoch 5/10
107668/107668 - 81s - loss: 0.7511 - accuracy: 0.7771 - val_loss: 0.8347
- val_accuracy: 0.7701
Epoch 6/10
107668/107668 - 81s - loss: 0.7061 - accuracy: 0.7872 - val_loss: 0.8329
- val_accuracy: 0.7676
Epoch 7/10
107668/107668 - 85s - loss: 0.6712 - accuracy: 0.7951 - val_loss: 0.8389
- val_accuracy: 0.7657
Epoch 8/10
107668/107668 - 86s - loss: 0.6437 - accuracy: 0.8014 - val_loss: 0.8562
- val_accuracy: 0.7635
Epoch 9/10
107668/107668 - 86s - loss: 0.6216 - accuracy: 0.8058 - val_loss: 0.8719
- val_accuracy: 0.7672
Epoch 10/10
107668/107668 - 86s - loss: 0.5980 - accuracy: 0.8101 - val_loss: 0.8638
- val_accuracy: 0.7652
33647/33647 - 5s - loss: 0.8659 - accuracy: 0.7634
Test loss: 0.8659017292210620
Test accuracy: 0.76339644
```

Gambar 7.204 Hasil Soal 12.

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

```
# In[13]:try various model configurations and parameters to find the best
# Melakukan import library time
import time
#Menginisiasi variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # Menginisiasi variabel model Sequential
            model = Sequential()
            #membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
            for i in range(conv2d_count):
                # syarat jika i samadengan bobotnya 0
                if i == 0:
                    # Penambahan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
                    # jika tidak
                else:
                    # Penambahan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
```

```
24                  # Penambahan method add pada variabel model
    dengan isian method  Max pooling berdimensi 2 dengan ukuran
    fixcel 2 x 2.
25                  model.add(MaxPooling2D(pool_size=(2, 2)))
26              # merubah feature gambar menjadi 1 dimensi vektor
27              model.add(Flatten())
28              # Penambahan method dense untuk pemadatan data dengan
     ukuran dense di tentukan dengan rumus fungsi tanh.
29              model.add(Dense(dense_size, activation='tanh'))
30              # membuat ketentuan jika pemangkasan lebih besar dari
     0 persen
31              if dropout > 0.0:
32                  # Penambahan method dropout pada model dengan
    nilai dari dropout
33                  model.add(Dropout(dropout))
34                  # Penambahan method dense dengan fungsi num
    classs dan rumus softmax
35              model.add(Dense(num_classes, activation='softmax'))
36              # mongkompile variabel model dengan hasi loss
    optimasi dan akurasi matrix
37              model.compile(loss='categorical_crossentropy',
    optimizer='adam', metrics=['accuracy'])
38              # melakukan log pada dir
39              log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' %
    (conv2d_count, dense_size, dropout)
40              # Menginisiasi variabel tensorboard dengan isian dari
     library keras dan nilai dari lig dir
41              tensorboard = keras.callbacks.TensorBoard(log_dir=
    log_dir)
42              # Menginisiasi variabel start dengan isian dari
    library time menggunakan method time
43
44              start = time.time()
45              # Penambahan method fit pada model dengan data dari
    train input train output nilai batch nilai epoch verbose
    nilai 20 persen validation split dan callback dengan nilai
    tnsorboard.
46              model.fit(train_input, train_output, batch_size=32,
    epochs=10,
47                        verbose=0, validation_split=0.2, callbacks
    =[tensorboard])
48              # Menginisiasi variabel score dengan nilai evaluasi
    dari model menggunakan data tes input dan tes output
49              score = model.evaluate(test_input, test_output,
    verbose=2)
50              # Menginisiasi variabel end
51              end = time.time()
52              # Menginisiasi variabel elapsed
53              elapsed = end - start
54              # mencetak hasil perhitungan
55              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))
56              results.append((conv2d_count, dense_size, dropout,
    score[0], score[1], elapsed))
```

```
...:                log_dir = r'logs\fit\%d_%d_%d_%d_%d_%d_%d_%d' 
% (conv2d_count, dense_size, dropout)
...:                # Menginisiasi variabel tensorboard dengan isian
dari library keras dan nilai dari lig dir
...:                tensorboard =
keras.callbacks.TensorBoard(log_dir=log_dir)
...:                # Menginisiasi variabel start dengan isian dari
library time menggunakan method time
...:
...:                start = time.time()
...:                # Penambahan method fit pada model dengan data dar
train input train output nilai batch nilai epoch verbose nilai 20 perse
validation split dan callback dengan nilai tensorboard.
...:                model.fit(train_input, train_output, batch_size=32
epochs=10,
...:                          verbose=0, validation_split=0.2,
callbacks=[tensorboard])
...:                # Menginisiasi variabel score dengan nilai evaluas
dari model menggunakan data tes input dan tes output
...:                score = model.evaluate(test_input, test_output,
verbose=2)
...:                # Menginisiasi variabel end
...:                end = time.time()
...:                # Menginisiasi variabel elapsed
...:                elapsed = end - start
...:                # mencetak hasil perhitungan
...:                print("Conv2D count: %d, Dense size: %d, Dropout: %
2f - Loss: %.2f, Accuracy: %.2f, Time: %d sec" % (conv2d_count,
dense_size, dropout, score[0], score[1], elapsed))
...:                results.append((conv2d_count, dense_size, dropout,
score[0], score[1], elapsed))
```

Gambar 7.205 Hasil Soal 13.

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

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

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_10 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
flatten_8 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_16 (Dense)             (None, 128)               147584
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_17 (Dense)             (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.206 Hasil Soal 14.

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

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

```
Epoch 1/10
168233/168233 - 83s - loss: 1.8214 - accuracy: 0.5784
Epoch 2/10
168233/168233 - 81s - loss: 1.0979 - accuracy: 0.7021
Epoch 3/10
168233/168233 - 77s - loss: 0.9910 - accuracy: 0.7246
Epoch 4/10
168233/168233 - 76s - loss: 0.9320 - accuracy: 0.7378
Epoch 5/10
168233/168233 - 79s - loss: 0.8945 - accuracy: 0.7453
Epoch 6/10
168233/168233 - 81s - loss: 0.8649 - accuracy: 0.7511
Epoch 7/10
168233/168233 - 78s - loss: 0.8446 - accuracy: 0.7553
Epoch 8/10
168233/168233 - 75s - loss: 0.8256 - accuracy: 0.7598
Epoch 9/10
168233/168233 - 74s - loss: 0.8111 - accuracy: 0.7616
Epoch 10/10
168233/168233 - 79s - loss: 0.8015 - accuracy: 0.7650
```

Gambar 7.207 Hasil Soal 15.

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

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

```
In [80]: model.save("mathsymbols.model")
WARNING:tensorflow:From C:\Users\TIKOMDIK 01\AppData\Roaming\Python
\Python37\site-packages\tensorflow_core\python\ops
\resource_variable_ops.py:1786: calling BaseResourceVariable.__init__
(from tensorflow.python.ops.resource_variable_ops) with constraint is
deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: mathsymbols.model\assets
```

Gambar 7.208 Hasil Soal 16.

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

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

```
In [81]: np.save('classes.npy', label_encoder.classes_)
```

Gambar 7.209 Hasil Soal 17.

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

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

```
In [80]: model.save("mathsymbols.model")
WARNING:tensorflow:From C:\Users\TIKOMDIK 01\AppData\Roaming\Python
\Python37\site-packages\tensorflow_core\python\ops
\resource_variable_ops.py:1786: calling BaseResourceVariable.__init__
(from tensorflow.python.ops.resource_variable_ops) with constraint is
deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
INFO:tensorflow:Assets written to: mathsymbols.model\assets
```

Gambar 7.210 Hasil Soal 18.

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

```
# In[19]:restore the class name to integer encoder
# Menginisiasi variabel label encoder ke 2 dengan isian fungsi
    label encoder.
label_encoder2 = LabelEncoder()
# Penambahan method classess dengan data classess yang di eksport
    tadi
label_encoder2.classes_ = np.load('classes.npy')
# membuat fumgsi predict dengan path img
def predict(img_path):
```

```
# Menginisiasi variabel newimg dengam membuay immage menjadi array dan membuka data berdasarkan img path
newimg = keras.preprocessing.image.img_to_array(pil_image.open(img_path))
# membagi data yang terdapat pada variabel newimg sebanyak 255
newimg /= 255.0

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

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

```
In [83]: label_encoder2 = LabelEncoder()
    ...: # Penambahan method classess dengan data classess yang di
eksport tadi
    ...: label_encoder2.classes_ = np.load('classes.npy')
    ...: # membuat fungsi predict dengan path img
    ...: def predict(img_path):
    ...:     # Menginisiasi variabel newimg dengam membuay immage menjad
array dan membuka data berdasarkan img path
    ...:     newimg =
keras.preprocessing.image.img_to_array(pil_image.open(img_path))
    ...:     # membagi data yang terdapat pada variabel newimg sebanyak
255
    ...:     newimg /= 255.0
    ...:
    ...:     # do the prediction
    ...:     # Menginisiasi variabel predivtion dengan isian variabel
model2 menggunakan fungsi predic dengan syarat variabel newimg dengan
data reshape
    ...:     prediction = model2.predict(newimg.reshape(1, 32, 32, 3))
    ...:
    ...:     # figure out which output neuron had the highest score, and
reverse the one-hot encoding
    ...:     # Menginisiasi variabel inverted  denagan label encoder2 da
menggunakan argmax untuk mencari skor luaran tertinggi
    ...:     inverted =
label_encoder2.inverse_transform([np.argmax(prediction)])
    ...:     # mencetak prediksi gambar dan confidence dari gambar.
    ...:     print("Prediction: %s, confidence: %.2f" % (inverted[0],
np.max(prediction)))
```

Gambar 7.211 Hasil Soal 19.

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

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

Gambar 7.212 Hasil Soal 20.

7.9.3 Penanganan Error

1. ModuleNotFoundError: No module named 'keras'

Gambar 7.213 NameError

2. Tuliskan Kode Error dan Jenis Error
   - ModuleNotFoundError
3. Cara Penangan Error
   - ModuleNotFoundError
     Error terjadi karena belum dilakukan instalasi keras pada komputer. Sehingga library keras tidak dapat dipanggil. Untuk mengatasinya, lakukan instalasi library keras terlebih dahulu pada komputer.

7.9.4 Bukti Tidak Plagiat

Gambar 7.214 Bukti Tidak Melakukan Plagiat Chapter 7

7.10 1174054 - Aulyardha Anindita

7.10.1 Teori

1. Jelaskan kenapa file teks harus dilakukan tokenizer. dilengkapi dengan ilustrasi atau gambar
   Kata pada suatu teks disebut token. Proses vektorisasi dari bentuk kata kedalam token biasa disebut dengan tokenizer dan tokenizer tersebut akan merubah suatu

teks tersebut menjadi simbol, kata, ataupun biner dan bentuk lainnya kedalam token.

Gambar 7.215 Tokenizer

2. Jelaskan konsep dasar K Fold Cross Validation pada dataset komentar Youtube pada kode listing 7.1.dilengkapi dengan ilustrasi atau gambar.
   StartifieldKFold berisi presentasi sampel dalam setiap kelas, dimana pada ilustrasi ini sampel tersebut dibagi menjadi 5 bagian dalam setiap classnya, lalu sampel tersebut dimasukkan kedalam class dari dataset youtube tersebut.

Gambar 7.216 K-Fold Cross Validation

3. Jelaskan apa maksudnya kode program for train, test in splits.dilengkapi dengan ilustrasi atau gambar
   Maksud dari kode program for train, test in splits, yaitu menguji apakah setiap data pada dataset sudah displit dan tidak terjadi penumpukan, yang dimana setiap class tidak akan muncul dengan id yang sama. misalnya ketika kita akan mempunyai 4 tas dengan model yang berbeda, lalu kita membaginya menjadi dua anak, tentunya setiap anak tersebut akan menerima tas yang modelnya tidak sama.

Gambar 7.217 Maksud Kode dor train, test in splits

4. Jelaskan apa maksudnya kode program train content = d[CONTENT].iloc[train idx] dan test content = d[CONTENT].iloc[test idx]. dilengkapi dengan ilustrasi atau gambar
   Maksud dari kode program tersebut adalah mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train idx dan test idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memilihnya untuk ditampilkan pada kolom yang diinginkan.

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

Gambar 7.218 Maksud Kode train content

5. Jelaskan apa maksud dari fungsi tokenizer = Tokenizer(num words=2000) dan tokenizer.fit on texts(train content), dilengkapi dengan ilustrasi atau gambar
   Tokenizer(num words=2000) digunakan untuk membaca kalimat yang telah dibuat menjadi token sebanyak 2000 kata tokenizer.fit on texts(train content) digunakan untuk membuat membaca data token teks yang telah dimasukkan kedalam fungsi yaitu fungsi train konten

Gambar 7.219 Maksud dari fungsi Tokenizer

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, mod='tfidf'), dilengkapi dengan ilustrasi kode atau gambar
   Maksudnya yaitu untuk variabel d train inputs akan melakukan tokenizer dari bentuk teks kematrix dari data train content dengan mode matriksnya yaitu tfidf begitu juga dengan variabel d test inputs untuk data test. berikut gambar ilustrasinya:

Gambar 7.220 Maksud Kode train inputs

7. Jelaskan apa maksud dari fungsi d train inputs = d train inputs/np.amax(np.absolute(d train dan d test inputs = d test inputs/np.amax(np.absolute(d test inputs)), dilengkapi dengan ilustrasi atau gambar.

Fungsi tersebut akan membagi matrix tfidf tadi dengan amax yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Yang hasilnya akan dimasukkan kedalam variabel d train inputs untuk data train dan d test inputs untuk data test dengan nominal absolut atau tanpa ada bilangan negatif dan koma.

Gambar 7.221 Maksud Kode train inputs

8. Jelaskan apa maksud fungsi dari d train outputs = np utils.to categorical(d[CLASS].iloc[tra
dan d test outputs = np utils.to categorical(d[CLASS].iloc[test idx]) dalam kode program, dilengkapi dengan ilustrasi atau gambar.
Dalam variabe ld train output dan d test outputs akan dilakukan one hot encoding,dimana np utils akan mengubah vektor dengan bentuk integer ke matriks kelas biner untuk kolom CLASS dimana nantinya hanya akan ada dua pilihan yaitu 1 atau 0. 1 untuk spam 0 untuk non spam atau sebaliknya. Berikut gambar ilustrasinya :

Gambar 7.222 Maksud Kode train outputs

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

- melakukan pemodelan sequential
- Layer pertama dense dari 512 neuron untuk inputan dengan inputan tadi yang sudah dijadikan matriks sebanyak 2000
- Activationnya menggunakan fungsi relu yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.

- Dropout ini untuk melakukan pembobotan, dimana pembobotan hanya dilakukan 50 persen saja agar tidak terjadi penumpukan data dari dense inputan tadi
- Dense 2 mengkategorikan 2 neuron untuk output nya yaitu 1 dan 0.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax.

Untuk ilustrasinya bisa dilihat pada gambar berikut :

Gambar 7.223 Ilustrasi Neural Network

10. Jelaskan apa maksud dari fungsi di listing 7.3 dengan parameter tersebut.
    Melakukan peng compile-an dari model Sequential tadi dengan Loss yan dengan merupakan fungsi optimisasi skor menggunakan categorical crossentropy,dan menggunakan algoritma adam sebagai optimizer. Adam yaitu algoritma pengoptimalan yang dapat digunakan sebagai ganti dari prosedur penurunan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training.Dengan metrik yaitu fungsi yang digunakan untuk menilai kinerja mode Anda disini menggunakan fungsi accuracy.

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

12. Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning
    Deep Neural Network adalah jaringan syaraf tiruan (JST) dengan beberapa lapisan antara lapisan input dan output. DNN menemukan manipulasi matematis yang benar untuk mengubah input menjadi output, apakah itu hubungan linear Deep Neural Network adalah jaringan syaraf tiruan (JST) dengan beberapa lapisan antara lapisan input dan output. DNN menemukan manipulasi matematis yang benar untuk mengubah input menjadi ouput, apakah itu hubungan linear atau hubungan non-linear. Merupakan jaringan syaraf dengan tingkat kompleksitas tertentu, jaringan syaraf dengan lebih dari dua lapisan. Deep Neural Network menggunakan pemodelan matematika yang canggih untuk memproses data dengan cara yang kompleks. DNN hanya terdiri dari dua lapisan yaitu input dan output, sedangkan dalam Deep learning kita dapat mendefinisikan layer sebanyak yang kita inginkan atau butuhkan.

13. Jelaskan dengan ilustrasi gambar buatan sendiri (langkah per langkah) bagaimana perhitungan algoritma kobvolasi dengan ukuran stride (NPM mod3+1) x (NPM mod3+1) yang terdapat pada max pooling
Karena hasil dari (NPM mod3+1) x (NPM mod3+1) = 2 x 2 = 4, berarti ada 4 metode max pooling, sebelum membuat ilustrasi perlu diketahui apa itu stride, stride adalag acuan atau parameter yang menentukan pergeseran pada filter fixcel, sebagai contoh nilai stride 1 yang berarti filter akan bergeser sebanyak satu fixcel secara vertikal dan horizontal. selanjutnya apa itu max pooling contoh pada suatu gambar ditentukan max pooling dari 3x3 dengan stride 1 yang berarti setiap pergeseran 1 pixcel akan diambil nilai terbesar dari pixcel 3 x 3 tersebut.

Gambar 7.224 Illustrasi perhitungan stride 1 max pooling

7.10.2 Praktek

1. Nomor 1

```
# In[1]:import lib
# mengimport libtary CSV untuk mengolah data ber ekstensi csv
import csv
#kemudian mengimport library Image yang berguna untuk mengolah
    data berupa gambar
from PIL import Image as pil_image
# kemudian mwngimport library keras yang menggunakan method
    preprocessing yang digunakan untuk membuat neural network
import keras.preprocessing.image
```

Dan hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.225 Hasil Nomor 1

2. Nomor 2

```
# In[2]:load all images (as numpy arrays) and save their classes
#Menginisiasi variabel imgs dan classes dengan variabel array kosong
imgs = []
classes = []
#membuka file hasy-data-labels.csv yang berada di folede HASYv2 yang di inisialisasi menjadi csvfile
with open('HASYv2/hasy-data-labels.csv') as csvfile:
    #Menginisiasi variabel csvreader yang berisi method csv.reader yang membaca variabel csvfile
    csvreader = csv.reader(csvfile)
    # Menginisiasi variabel i dengan isi 0
    i = 0
    # membuat looping pada variabel csvreader
    for row in csvreader:
        # dengan ketentuan jika i lebihkecil daripada o
        if i > 0:
            # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            img = keras.preprocessing.image.img_to_array(pil_image.open("HASYv2/" + row[0]))
            #Pembagian data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # Penambahan nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # Penambahan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

Untuk hasil plotnya bisa dilihat pada gambar berikut:

Gambar 7.226 Hasil Nomor

3. Nomor 3

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

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

Hasilnya dapat dilihat pada gambar berikut:

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

Gambar 7.227 Hasil Nomor 3

4. Nomor 4

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

Untuk hasilnya bisa dilihat pada gambar berikut :

```
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 7.228 Hasil Nomor 4

5. Nomor 5

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

```
# Melakukan import library OneHotEncoder dari sklearn
from sklearn.preprocessing import OneHotEncoder
```

Untuk hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.229 Hasil Nomor 5

6. Nomor 6

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

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

Hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.230 Hasil Nomor 6

7. Nomor 7

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

Hasilnya bisa dilihat pada gambar berikut :

```
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 7.231 Hasil Nomor 7

8. Nomor 8

```
# In[8]:convert train and test output to one-hot
# Menconvert data train output  mengguanakn variabel
    label_encoder kedalam variabel train_output_int
train_output_int = label_encoder.transform(train_output)
# Menconvert variabel train_output_int kedalam fungsi
    onehot_encoder
train_output = onehot_encoder.transform(train_output_int.reshape(
    len(train_output_int), 1))
# Menconvert data test_output mengguanakn variabel label_encoder
    kedalam variabel test_output_int
test_output_int = label_encoder.transform(test_output)
# Menconvert variabel test_output_int kedalam fungsi
    onehot_encoder
test_output = onehot_encoder.transform(test_output_int.reshape(
    len(test_output_int), 1))
# Menginisiasi variabel num_classes dengan isi variabel
    label_encoder dan classess
num_classes = len(label_encoder.classes_)
# mencetak hasil dari nomer Class beruapa persen
print("Number of classes: %d" % num_classes)
```

Untuk hasilnya bisa dilihat pada gambar berikut :

```
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 7.232 Hasil Nomor 8

9. Nomor 9

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

Hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.233 Hasil Nomor 9

10. Nomor 10

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

Hasilnya bisa dilihat pada gambar berikut :

```
...: 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 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (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 7.234 Hasil Nomor 10

11. Nomor 11

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

Hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.235 Hasil Nomor 11

12. Nomor 12

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

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

Hasilnya bisa dilihat pada gambar berikut :

```
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
\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_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 7.236 Hasil Nomor 12

13. Nomor 13

```
# In[13]:try various model configurations and parameters to find the best
# Melakukan import library time
import time
#Menginisiasi variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
        # membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
        for dropout in [0.0, 0.25, 0.50, 0.75]:
            # Menginisiasi variabel model Sequential
            model = Sequential()
            #membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
            for i in range(conv2d_count):
                # syarat jika i samadengan bobotnya 0
                if i == 0:
                    # Penambahan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))
                    # jika tidak
                else:
                    # Penambahan method add pada variabel model dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan fungsi aktivasi relu
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu'))
                # Penambahan method add pada variabel model dengan isian method  Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
                model.add(MaxPooling2D(pool_size=(2, 2)))
            # merubah feature gambar menjadi 1 dimensi vektor
            model.add(Flatten())
            # Penambahan method dense untuk pemadatan data dengan ukuran dense di tentukan dengan rumus fungsi tanh.
            model.add(Dense(dense_size, activation='tanh'))
            # membuat ketentuan jika pemangkasan lebih besar dari 0 persen
            if dropout > 0.0:
                # Penambahan method dropout pada model dengan nilai dari dropout
                model.add(Dropout(dropout))
                # Penambahan method dense dengan fungsi num classs dan rumus softmax
            model.add(Dense(num_classes, activation='softmax'))
            # mongkompile variabel model dengan hasi loss optimasi dan akurasi matrix
            model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

```
        # melakukan log pada dir
        log_dir = './logs/conv2d_%d-dense_%d-dropout_%.2f' % (conv2d_count, dense_size, dropout)
        # Menginisiasi variabel tensorboard dengan isian dari library keras dan nilai dari lig dir
        tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir)
        # Menginisiasi variabel start dengan isian dari library time menggunakan method time

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

Hasilnya bisa dilihat pada gambar berikut :

```
...: 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
```

Gambar 7.237 Hasil Nomor 13

14. Nomor 14

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

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

Hasilnya bisa dilihat pada gambar berikut :

```
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 (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 7.238 Hasil Nomor 14

15. Nomor 15

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

Hasilnya bisa dilihat pada gambar berikut :

```
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 7.239 Hasil Nomor 15

16. Nomor 16

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

Hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.240 Hasil Nomor 16

17. Nomor 17

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

Hasilnya bisa dilihat pada gambar berikut :

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

Gambar 7.241 Hasil Nomor 17

18. Nomor 18

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

Hasilnya bisa dilihat pada gambar berikut :

```
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 7.242 Hasil Nomor 18

19. Nomor 19

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

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

```
16
17      # figure out which output neuron had the highest score, and
        reverse the one-hot encoding
18      # Menginisiasi variabel inverted  denagan label encoder2 dan
         menggunakan argmax untuk mencari skor luaran tertinggi
19      inverted = label_encoder2.inverse_transform([np.argmax(
        prediction)])
20      # mencetak prediksi gambar dan confidence dari gambar.
21      print("Prediction: %s, confidence: %.2f" % (inverted[0], np.
        max(prediction)))
```

Hasilnya bisa dilihat pada gambar berikut :

```
In [25]: label_encoder2 = LabelIncoder()
    ...: 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 7.243 Hasil Nomor 19

20. Nomor 20

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

Hasilnya bisa dilihat pada gambar berikut :

```
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 7.244 Hasil Nomor 20

7.10.3 Penanganan Error

1. ScreenShoot Error

```
  File "C:\ProgramData\Anaconda3\lib\site-packages\keras\backend
\tensorflow_backend.py", line 5, in <module>
    import tensorflow as tf

ModuleNotFoundError: No module named 'tensorflow'
```

Gambar 7.245 Module Not Found Error

2. Tuliskan Kode Error dan Jenis Error
   - Module Not Found Error
3. Cara Penanganan Error
   - Module Not Found Error
     Error terjadi karena belum menginstal package atau library tensorflow Untuk mengatasinya dengan mengisntal library tensorflow pada anaconda

7.10.4 Bukti Tidak Plagiat

Gambar 7.246 Bukti Plagiasrisme

7.10.5 Link Youtube

https://youtu.be/5jv7jHYu7pg

7.11 1174083 - Bakti Qilan Mufid

Chapter 7 - CNN

7.11.1 Teori

7.11.1.1 Jelaskan kenapa file teks harus dilakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.

Karena untuk memudahkan mesin memahami maksud dari apa yang usr inginkan. Dalam teks file, kata disebut token, dan proses vektorisasi dari bentuk kata kedalam bentuk token tersebut disebut tokenizer dan tokenizer akan merubah sebuah teks menjadi simbol, kata, ataupun biner dan bentuk lainya kedalam token. Untuk ilustrasinya bisa seperti berikut, terdapat sebuah kalimat a=yang akan dilakukan tokenizer kalimatnya: ”jangan panggil aku anak kecil paman” maka ketika dilakukan tokenizer hasilnya akan menjadi [’jangan’,’panggil’,’aku’,’anak’,’kecil’,’paman’].

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

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

Listing 7.7 K-fold Cross Validation

Konsep sederhana dari K-Fold Cross Validation pada code tesebut ialah terdapat kfold yang bertujuan untuk melakukan split data menjadi 5 bagian dari dataset komentar Youtube tersebut. Sehingga dari setiap data yang sudah dibagi tersebut akan menghasilkan presentase dari setiap bagiannya, untuk menghasilkan hasil akhir dengan presentase yang cukup baik. Untuk ilustrasi, lihat gambar berikut:

Gambar 7.247 gambaran penjelasan no. 2

7.11.1.3 Jelaskan apa maksudnya kode program for train, test in splits. dilengkapi dengan gambar atau ilustrasi

Untuk penjelasan nya yaitu For train berfungsi untuk membagi data tersebut menjadi data training. Sedangkan test in splits berfungsi untuk menguji apakah dataset tersebut sudah dibagi menjadi beberapa bagian atau masih menumpuk.

Gambar 7.248 gambaran penjelasan no. 3

7.11.1.4 jelaskan maksudnya kode program

Maksudnya yaitu mengambil data pada kolom atau index CONTENT yang merupakan bagian dari train_idx dan test_idx. Ilustrasinya, ketika data telah diubah menjadi train dan test maka kita dapat memil-

ihnya untuk ditampilkan pada kolom yang diinginkan. untuk ilustrasinya ada pada kode berikut:

```
import pandas as pd

mydict = [{'w': 6, 'x': 7, 'y': 8, 'z': 9},
          {'w': 600, 'x': 700, 'y': 800, 'z': 900},
          {'w': 6000, 'x': 7000, 'y': 8000, 'z': 9000 }]
df = pd.DataFrame(mydict)
df

type(df.iloc[0])
df.iloc[0]
```

Gambar 7.249 gambaran penjelasan no. 4

7.11.1.5 Jelaskan apa maksud dari fungsi dilengkapi dengan ilustrasi atau gambar

Fungsi tokenizer ini berfungsi untuk melakukan vektorisasi data kedalam bentuk token sebanyak 2000 kata. Dan selanjutnya akan melakukan fit tokenizer hanya untuk data training saja tidak dengan data testingnya. Untuk ilustrasi lihat gambar berikut:

Gambar 7.250 gambaran penjelasan no. 5

7.11.1.6 Jelaskan apa maksud dari dilengkapi dengan ilustrasi kode dan atau gambar

Maksudnya yaitu untuk variabel d_train_inputs akan melakukan tokenizer dari bentuk teks ke matrix dari data train_content dengan mode matriksnya yaitu tfidf begitu juga dengan variabel d_test_inputs untuk data test.

Gambar 7.251 gambaran penjelasan no. 6

7.11.1.7 Jelaskan apa maksud dari fungsi dilengkapi dengan ilustrasi atau gambar

Fungsi tersebut akan membagi matrix tfidf tadi dengan np.amax yaitu mengembalikan maksimum array atau maksimum sepanjang sumbu. Jika sumbu tidak ada, hasilnya adalah nilai skalar. Jika sumbu diberikan, hasilnya adalah array dimensi a.ndim - 1. Yang hasilnya akan dimasukan kedalam variabel d_train_inputs untuk data train dan d_test_inputs untuk data_test dengan nominal absolut atau tanpa ada bilangan negatif dan koma.

```
>>> import numpy as np
>>> x = np.array([-5, 6, 1.2])
>>> np.amax(np.absolute(x))
6.0
>>> x = np.array([-5, 6, 1.2, -8])
>>> np.amax(np.absolute(x))
8.0
```

Gambar 7.252 gambaran penjelasan no. 7

7.11.1.8 Jelaskan apa maksud dari dilengkapi dengan ilustrasi atau gambar

Dalam variabel d_train_output dan d_test_outputs akan dilakukan one-hot-encoding, dimana np_utils akan mengubah vektor dengan bentuk integer ke matriks kelas biner untuk kolom CLASS dimana nantinya hanya akan ada dua pilihan yaitu 1 atau 0. 1 untuk spam 0 untuk non spam atau sebaliknya.

```
>>> bakti = np.array([0,2,1,2,0])
>>> to_categorical(bakti)
array([[1., 0., 0.],
       [0., 0., 1.],
       [0., 1., 0.],
       [0., 0., 1.],
       [1., 0., 0.]], dtype=float32)
>>>
```

Gambar 7.253 gambaran penjelasan no. 8

7.11.1.9 Jelaskan apa maksud dari fungsi di listing 7.2. Gambar Neural Networknya dari model kode tersebut

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

Listing 7.8 model Neural Network

Penjelasannya sebagai berikut :

- Melakukan pemodelan Sequential

- Layer pertama dense dari 512 neuron untuk inputan dengan inputan tadi yang sudah dijadikan matriks sebanyak 2000
- Activationnya menggunakan fungsi 'relu' yaitu jika ada inputan dengan nilai maksimum maka inputan itu yang akan terpilih.
- Dropout ini untuk melakukan pembobotan, dimana pembobotan hanya dilakukan 50% saja agar tidak terjadi penumpukan data dari dense inputan tadi
- Dense 2 mengkategorikan 2 neuron untuk output nya yaitu 1 dan 0.
- Untuk dense diatas aktivasinya menggunakan fungsi Softmax.

Gambar 7.254 gambaran penjelasan no. 9

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

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

Listing 7.9 compile model

Melakukan peng compile-an dari model Sequential tadi dengan Loss yandengang merupakan fungsi optimisasi skor menggunakan categorical crossentropy , dan menggunakan algoritma adam sebagai optimizer. Adam yaitu algoritma pengoptimalan yang dapat digunakan sebagai ganti dari prosedur penurunan gradien stokastik klasik untuk memperbarui bobot jaringan yang berulang berdasarkan data training.Dengan metrik yaitu fungsi yang digunakan untuk menilai kinerja mode Anda disini menggunakan fungsi accuracy.

Gambar 7.255 gambaran penjelasan no. 9

7.11.1.11 Jelaskan apa itu Deep Learning

Deep Learning adalah subbidang machine learning yang berkaitan dengan algoritma yang terinspirasi oleh struktur dan fungsi otak yang disebut jaringan saraf tiruan atau Artificial Neural Networks. Jaringan saraf tiruan, algoritma yang terinspirasi oleh otak manusia, belajar dari sejumlah besar data. Demikian pula dengan bagaimana kita belajar dari pengalaman, algoritma pembelajaran yang mendalam akan melakukan tugas berulang kali, setiap kali sedikit mengubahnya untuk meningkatkan hasilnya.

7.11.1.12 Jelaskan apa itu Deep Neural Network, dan apa bedanya dengan Deep Learning

Deep Neural Network adalah jaringan syaraf tiruan (JST) dengan beberapa lapisan antara lapisan input dan output. DNN menemukan manipulasi matematis yang benar untuk mengubah input menjadi output, apakah itu hubungan linear atau hubungan non-linear. Merupakan jaringan syaraf dengan tingkat kompleksitas tertentu, jaringan syaraf dengan lebih dari dua lapisan. Deep Neural Network menggunakan pemodelan matematika yang canggih untuk memproses data dengan cara yang kompleks.

DNN hanya terdiri dari dua laipsan yaitu input dan output, sedangkan dalam Deep learning kita dapat mendefiniskan layer sebanyak yang kita inginkan atau butuhkan.

7.11.1.13 Jelaskan dengan ilustrasi gambar buatan sendiri(langkah per langkah) bagaimana perhitungan algoritma konvolusi dengan ukuran stride (NPM mod 3+1) x (NPM mod 3+1) yang terdapat max pooling

Stride adalah parameter yang menentukan berapa jumlah pergeseran filter. stride = (NPM mod 3+1) x (NPM mod 3+1) = 1

- disini saya mempunyai data seperti berikut:

Gambar 7.256 gambaran penjelasan no. 9

- Kemudian hitung konvolusi untuk setiap matriksnya seperti berikut :
  - pertama

Gambar 7.257 Algoritma Konvolusi

  - kedua

Gambar 7.258 Algoritma Konvolusi

  - ketiga

Gambar 7.259 Algoritma Konvolusi

– keempat

Gambar 7.260 Algoritma Konvolusi

– kelima

Gambar 7.261 Algoritma Konvolusi

– keenam

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

x

0	2	-1
-1	0	2
0	1	1

=

	15		

Gambar 7.262 Algoritma Konvolusi

– ketujuh

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

x

0	2	-1
-1	0	2
0	1	1

=

		11	

Gambar 7.263 Algoritma Konvolusi

– kedelapan

1	2	5	2	1	4
3	2	5	1	1	5
3	2	4	1	5	3
5	3	5	1	3	5
5	1	1	5	5	1
5	5	4	3	3	1

x

0	2	-1
-1	0	2
0	1	1

=

			10

Gambar 7.264 Algoritma Konvolusi

– kesembilan

Gambar 7.265 Algoritma Konvolusi

– kesepuluh

Gambar 7.266 Algoritma Konvolusi

– kesebelas

Gambar 7.267 Algoritma Konvolusi

– keduabelas

Gambar 7.268 Algoritma Konvolusi

– ketigabelas

Gambar 7.269 Algoritma Konvolusi

– keempatbelas

Gambar 7.270 Algoritma Konvolusi

– kelimabelas

Gambar 7.271 Algoritma Konvolusi

– keenambelas

Gambar 7.272 Algoritma Konvolusi

- Didapatkan hasil akhir nilai konvolusi dan juga max poolingnya seperti berikut:

Gambar 7.273 Hasil akhir nilai konvolusi dan max pooling

7.11.2 Praktek

7.11.2.1 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

Penjelasan kode:

```
# In[1]:import lib
import csv #Import library csv
```

```
from PIL import Image as pil_image #Import library Image yaitu fungsi PIL (Python Imaging Library) yang berguna untuk mengolah data berupa gambar
import keras.preprocessing.image #Import library keras yang menggunakan method preprocessing yang digunakan untuk membuat neural network
```

Listing 7.10 Kode program blok(1)

Hasil Skrinsut:

Gambar 7.274 Hasil Skrinsut

7.11.2.2 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

Penjelasan kode:

```
# In[2]:load all images (as numpy arrays) and save their classes
imgs = [] #Membuat variabel imgs
classes = [] #Membuat variabel classes dengan variabel kosong
with open('HASYv2/hasy-data-labels.csv') as csvfile: #Membuka file hasy-data-labels.csv
    csvreader = csv.reader(csvfile) #Membuat variabel csvreader yang berisi method csv.reader untuk membaca csvfile
    i = 0 # membuat variabel i dengan isi 0
    for row in csvreader: # Membuat looping pada variabel csvreader
        if i > 0: #Ketentuannya jika i lebih kecil daripada 0
            img = keras.preprocessing.image.img_to_array(pil_image.open("N:/HASYv2/" + row[0])) #Dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai -1.0 dan 1.0
            # 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 #Membagi data yang ada pada fungsi img sebanyak 255.0
            imgs.append((row[0], row[2], img)) #Menambah nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
```

```
            classes.append(row[2]) #Menambahkan nilai pada row ke 2 pada variabel classes
        i += 1 #Menambah nilai satu pada variabel i
```

Listing 7.11 Kode program blok(2)

Hasil Skrinsut:

```
[ ] # In[2]:load all images (as numpy arrays) and save their classes

    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 7.275 Hasil Skrinsut

7.11.2.3 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

Penjelasan kode:

```
# In[3]:shuffle the data, split into 80% train, 20% test
import random #Mengimport library random
random.shuffle(imgs) #Melakukan randomize pada fungsi imgs
split_idx = int(0.8*len(imgs)) #Membuat variabel split_idx dengan nilai integer 80 persen dikali dari pengembalian jumlah dari variabel imgs
train = imgs[:split_idx] #Membuat variabel train dengan isi sebelum split idx
test = imgs[split_idx:] #Membuat variabel test dengan isi setelah split idx
```

Listing 7.12 Kode program blok(3)

Hasil Skrinsut:

```
[ ] # In[3]:shuffle the data, split into 80% train, 20% test

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

Gambar 7.276 Hasil Skrinsut

7.11.2.4 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

Penjelasan kode:

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

Listing 7.13 Kode program blok(4)

Hasil Skrinsut:

Gambar 7.277 Hasil Skrinsut

7.11.2.5 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

Penjelasan kode:

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

Listing 7.14 Kode program blok(5)

Hasil Skrinsut:

Gambar 7.278 Hasil Skrinsut

7.11.2.6 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

Penjelasan kode:

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

Listing 7.15 Kode program blok(6)

Hasil Skrinsut:

Gambar 7.279 Hasil Skrinsut

7.11.2.7 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

Penjelasan kode:

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

Listing 7.16 Kode program blok(7)

Hasil Skrinsut:

Gambar 7.280 Hasil Skrinsut

7.11.2.8 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

Penjelasan kode:

```
# In[8]:convert train and test output to one-hot
train_output_int = label_encoder.transform(train_output) #Mengkonvert data train output mengguanakn variabel label_encoder kedalam variabel train_output_int
train_output = onehot_encoder.transform(train_output_int.reshape(len(train_output_int), 1)) #Mengkonvert variabel train_output_int kedalam fungsi onehot_encoder
test_output_int = label_encoder.transform(test_output) #Mengkonvert data test_output mengguanakn variabel label_encoder kedalam variabel test_output_int
test_output = onehot_encoder.transform(test_output_int.reshape(len(test_output_int), 1)) #Mengkonvert variabel test_output_int kedalam fungsi onehot_encoder
num_classes = len(label_encoder.classes_) #Membuat variabel num_classes dengan isi variabel label_encoder dan classess
print("Number of classes: %d" % num_classes) #Mencetak hasil dari nomer Class beruapa persen
```

Listing 7.17 Kode program blok(8)

Hasil Skrinsut:

Gambar 7.281 Hasil Skrinsut

7.11.2.9 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

Penjelasan kode:

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

Listing 7.18 Kode program blok(9)

Hasil Skrinsut:

Gambar 7.282 Hasil Skrinsut

7.11.2.10 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

Penjelasan kode:

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

Listing 7.19 Kode program blok(10)

Hasil Skrinsut:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 6, 6, 32)          0
_________________________________________________________________
flatten (Flatten)            (None, 1152)              0
_________________________________________________________________
dense (Dense)                (None, 1024)              1180672
_________________________________________________________________
dropout (Dropout)            (None, 1024)              0
_________________________________________________________________
dense_1 (Dense)              (None, 369)               378225
=================================================================
Total params: 1,569,041
Trainable params: 1,569,041
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.283 Hasil Skrinsut

7.11.2.11 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

Penjelasan kode:

```
# In[11]: import sequential
import keras.callbacks #Mengimport library keras dengan fungsi
    callbacks
tensorboard = keras.callbacks.TensorBoard(log_dir='hasyv2/logs/mnist-
    style') #Membuat variabel tensorboard dengan isi lib keras
```

Listing 7.20 Kode program blok(11)

Hasil Skrinsut:

```
[ ] # In[11]: import sequential

    tensorboard = keras.callbacks.TensorBoard(log_dir='.\\logs\\mnist-style')
```

Gambar 7.284 Hasil Skrinsut

7.11.2.12 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

Penjelasan kode:

```
# In[12]: 5menit kali 10 epoch = 50 menit
model.fit(train_input, train_output, #Fungsi model ditambahkan fungsi
    fit untuk mengetahui perhitungan dari train_input train_output
          batch_size=32, #Dengan batch size 32 bit
          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])
```

Listing 7.21 Kode program blok(12)

Hasil Skrinsut:

```
Epoch 1/10
3365/3365 - 100s - loss: 1.5393 - accuracy: 0.6264 - val_loss: 0.9766 - val_accuracy: 0.7349
Epoch 2/10
3365/3365 - 102s - loss: 0.9562 - accuracy: 0.7345 - val_loss: 0.8892 - val_accuracy: 0.7492
Epoch 3/10
3365/3365 - 103s - loss: 0.8443 - accuracy: 0.7575 - val_loss: 0.8475 - val_accuracy: 0.7618
Epoch 4/10
3365/3365 - 103s - loss: 0.7695 - accuracy: 0.7730 - val_loss: 0.8456 - val_accuracy: 0.7639
Epoch 5/10
3365/3365 - 103s - loss: 0.7142 - accuracy: 0.7864 - val_loss: 0.8427 - val_accuracy: 0.7693
Epoch 6/10
3365/3365 - 104s - loss: 0.6754 - accuracy: 0.7931 - val_loss: 0.8405 - val_accuracy: 0.7702
Epoch 7/10
3365/3365 - 103s - loss: 0.6396 - accuracy: 0.8013 - val_loss: 0.8521 - val_accuracy: 0.7634
Epoch 8/10
3365/3365 - 103s - loss: 0.6130 - accuracy: 0.8073 - val_loss: 0.8747 - val_accuracy: 0.7610
Epoch 9/10
3365/3365 - 104s - loss: 0.5872 - accuracy: 0.8117 - val_loss: 0.8675 - val_accuracy: 0.7655
Epoch 10/10
3365/3365 - 103s - loss: 0.5673 - accuracy: 0.8176 - val_loss: 0.8994 - val_accuracy: 0.7681
1052/1052 - 9s - loss: 0.8918 - accuracy: 0.7703
Test loss: 0.8918401002883911
Test accuracy: 0.770321249961853
```

Gambar 7.285 Hasil Skrinsut

7.11.2.13 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

Penjelasan kode:

```
# In[13]:try various model configurations and parameters to find the best
import time #Mengimport library time
results = [] #Membuat variabel result dengan array kosong
for conv2d_count in [1, 2]: #Melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
    for dense_size in [128, 256, 512, 1024, 2048]: #Menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
        for dropout in [0.0, 0.25, 0.50, 0.75]: #Membuat looping untuk memangkas masing-masing data dengan ketentuan 0 persen 25 persen 50 persen dan 75 persen.
            model = Sequential() #Membuat variabel model Sequential
            for i in range(conv2d_count): #Membuat looping untuk variabel i dengan jarak dari hasil konvolusi.
                if i == 0: #Syarat jika i samadengan bobotnya 0
                    model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0]))) #Menambahkan method add pada variabel model dengan konvolusi 2 dimensi 32 bit didalamnya dan membuat kernel dengan ukuran 3 x 3 dan rumus aktifasi relu dan data shape yang di hitung dari data train.
```

```
11                  else: #Jika tidak
12                      model.add(Conv2D(32, kernel_size=(3, 3),
    activation='relu'))  #Menambahkan method add pada variabel model
    dengan konvolusi 2 dimensi 32 bit dengan ukuran kernel 3 x3 dan
    fungsi aktivasi relu
13                  model.add(MaxPooling2D(pool_size=(2, 2))) #
    Menambahkan method add pada variabel model dengan isian method
    Max pooling berdimensi 2 dengan ukuran fixcel 2 x 2.
14          model.add(Flatten()) #Merubah feature gambar menjadi 1
    dimensi vektor
15          model.add(Dense(dense_size, activation='tanh')) #
    Menambahkan method dense untuk pemadatan data dengan ukuran dense
     di tentukan dengan rumus fungsi tanh.
16          if dropout > 0.0: #Membuat ketentuan jika pemangkasan
    lebih besar dari 0 persen
17              model.add(Dropout(dropout)) #Menambahkan method
    dropout pada model dengan nilai dari dropout
18          model.add(Dense(num_classes, activation='softmax')) #
    Menambahkan method dense dengan fungsi num classs dan rumus
    softmax
19          model.compile(loss='categorical_crossentropy', optimizer=
    'adam', metrics=['accuracy']) #Mengcompile variabel model dengan
    hasi loss optimasi dan akurasi matrix
20          log_dir = 'hasyv2/logs/conv2d_%d-dense_%d-dropout_%.2f' %
     (conv2d_count, dense_size, dropout) #Melakukan log
21          tensorboard = keras.callbacks.TensorBoard(log_dir=log_dir
    ) # membuat variabel tensorboard dengan isian dari library keras
    dan nilai dari log_dir
22
23          start = time.time() #Membuat variabel start dengan isian
    dari library time menggunakan method time
24          model.fit(train_input, train_output, batch_size=32,
    epochs=10,
25                    verbose=0, validation_split=0.2, callbacks=[
    tensorboard]) #Menambahkan method fit pada model dengan data dari
     train input train output nilai batch nilai epoch verbose nilai
    20 persen validation split dan callback dengan nilai tensorboard.
26          score = model.evaluate(test_input, test_output, verbose
    =2) #Membuat variabel score dengan nilai evaluasi dari model
    menggunakan data tes input dan tes output
27          end = time.time() #Membuat variabel end
28          elapsed = end - start  #Membuat variabel elapsed
29          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)) #Mencetak
    hasil perhitungan
30          results.append((conv2d_count, dense_size, dropout, score
    [0], score[1], elapsed))
```

Listing 7.22 Kode program blok(13)

Hasil Skrinsut:

```
[ ] 1052/1052 - 6s - loss: 1.2295 - accuracy: 0.7344
    Conv2D count: 1, Dense size: 128, Dropout: 0.00 - Loss: 1.23, Accuracy: 0.73, Time: 556 sec
    1052/1052 - 6s - loss: 0.9560 - accuracy: 0.7614
    Conv2D count: 1, Dense size: 128, Dropout: 0.25 - Loss: 0.96, Accuracy: 0.76, Time: 574 sec
    1052/1052 - 7s - loss: 0.8298 - accuracy: 0.7737
    Conv2D count: 1, Dense size: 128, Dropout: 0.50 - Loss: 0.83, Accuracy: 0.77, Time: 567 sec
    1052/1052 - 7s - loss: 0.8140 - accuracy: 0.7693
    Conv2D count: 1, Dense size: 128, Dropout: 0.75 - Loss: 0.81, Accuracy: 0.77, Time: 592 sec
    1052/1052 - 7s - loss: 1.3661 - accuracy: 0.7432
    Conv2D count: 1, Dense size: 256, Dropout: 0.00 - Loss: 1.37, Accuracy: 0.74, Time: 722 sec
    1052/1052 - 7s - loss: 1.1462 - accuracy: 0.7532
    Conv2D count: 1, Dense size: 256, Dropout: 0.25 - Loss: 1.15, Accuracy: 0.75, Time: 726 sec
    1052/1052 - 7s - loss: 0.9360 - accuracy: 0.7685
    Conv2D count: 1, Dense size: 256, Dropout: 0.50 - Loss: 0.94, Accuracy: 0.77, Time: 728 sec
    1052/1052 - 7s - loss: 0.7998 - accuracy: 0.7783
    Conv2D count: 1, Dense size: 256, Dropout: 0.75 - Loss: 0.80, Accuracy: 0.78, Time: 734 sec
    1052/1052 - 9s - loss: 1.6735 - accuracy: 0.7346
    Conv2D count: 1, Dense size: 512, Dropout: 0.00 - Loss: 1.67, Accuracy: 0.73, Time: 1057 sec
    1052/1052 - 8s - loss: 1.5223 - accuracy: 0.7476
    Conv2D count: 1, Dense size: 512, Dropout: 0.25 - Loss: 1.52, Accuracy: 0.75, Time: 1049 sec
    1052/1052 - 9s - loss: 1.3185 - accuracy: 0.7458
    Conv2D count: 1, Dense size: 512, Dropout: 0.50 - Loss: 1.32, Accuracy: 0.75, Time: 1091 sec
    1052/1052 - 9s - loss: 0.8951 - accuracy: 0.7719
    Conv2D count: 1, Dense size: 512, Dropout: 0.75 - Loss: 0.90, Accuracy: 0.77, Time: 1046 sec
    1052/1052 - 13s - loss: 2.5109 - accuracy: 0.7134
    Conv2D count: 1, Dense size: 1024, Dropout: 0.00 - Loss: 2.51, Accuracy: 0.71, Time: 2283 sec
    1052/1052 - 14s - loss: 2.2605 - accuracy: 0.7204
    Conv2D count: 1, Dense size: 1024, Dropout: 0.25 - Loss: 2.26, Accuracy: 0.72, Time: 2213 sec
    1052/1052 - 13s - loss: 1.6548 - accuracy: 0.7309
    Conv2D count: 1, Dense size: 1024, Dropout: 0.50 - Loss: 1.65, Accuracy: 0.73, Time: 2222 sec
    1052/1052 - 13s - loss: 1.0859 - accuracy: 0.7440
    Conv2D count: 1, Dense size: 1024, Dropout: 0.75 - Loss: 1.09, Accuracy: 0.74, Time: 2182 sec
```

```
1052/1052 - 19s - loss: 2.2024 - accuracy: 0.7027
Conv2D count: 1, Dense size: 2048, Dropout: 0.00 - Loss: 2.20, Accuracy: 0.70, Time: 3958 sec
1052/1052 - 19s - loss: 2.2918 - accuracy: 0.6986
Conv2D count: 1, Dense size: 2048, Dropout: 0.25 - Loss: 2.29, Accuracy: 0.70, Time: 3974 sec
1052/1052 - 20s - loss: 1.7761 - accuracy: 0.7164
Conv2D count: 1, Dense size: 2048, Dropout: 0.50 - Loss: 1.78, Accuracy: 0.72, Time: 4153 sec
1052/1052 - 19s - loss: 1.2051 - accuracy: 0.7334
Conv2D count: 1, Dense size: 2048, Dropout: 0.75 - Loss: 1.21, Accuracy: 0.73, Time: 4160 sec
1052/1052 - 8s - loss: 0.8711 - accuracy: 0.7643
Conv2D count: 2, Dense size: 128, Dropout: 0.00 - Loss: 0.87, Accuracy: 0.76, Time: 704 sec
1052/1052 - 8s - loss: 0.7883 - accuracy: 0.7782
Conv2D count: 2, Dense size: 128, Dropout: 0.25 - Loss: 0.79, Accuracy: 0.78, Time: 709 sec
1052/1052 - 8s - loss: 0.7600 - accuracy: 0.7822
Conv2D count: 2, Dense size: 128, Dropout: 0.50 - Loss: 0.76, Accuracy: 0.78, Time: 718 sec
1052/1052 - 8s - loss: 0.8216 - accuracy: 0.7634
Conv2D count: 2, Dense size: 128, Dropout: 0.75 - Loss: 0.82, Accuracy: 0.76, Time: 730 sec
1052/1052 - 8s - loss: 1.0374 - accuracy: 0.7486
Conv2D count: 2, Dense size: 256, Dropout: 0.00 - Loss: 1.04, Accuracy: 0.75, Time: 746 sec
1052/1052 - 8s - loss: 0.8642 - accuracy: 0.7711
Conv2D count: 2, Dense size: 256, Dropout: 0.25 - Loss: 0.86, Accuracy: 0.77, Time: 754 sec
1052/1052 - 8s - loss: 0.7819 - accuracy: 0.7816
Conv2D count: 2, Dense size: 256, Dropout: 0.50 - Loss: 0.78, Accuracy: 0.78, Time: 755 sec
1052/1052 - 8s - loss: 0.7786 - accuracy: 0.7750
Conv2D count: 2, Dense size: 256, Dropout: 0.75 - Loss: 0.78, Accuracy: 0.78, Time: 755 sec
```

Gambar 7.286 Hasil Skrinsut

7.11.2.14 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

Penjelasan kode:

```
# In[14]:rebuild/retrain a model with the best parameters (from the search) and use all data
model = Sequential() #Membuat variabel model dengan isian library Sequential
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=np.shape(train_input[0])))#Variabel model di tambahkan library Conv2D tigapuluh dua bit dengan ukuran kernel 3 x 3 dan fungsi penghitungan relu dang menggunakan data train_input
```

```
model.add(MaxPooling2D(pool_size=(2, 2))) #Variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixel
model.add(Conv2D(32, (3, 3), activation='relu')) #Variabel model di tambahkan dengan library Conv2D 32bit dengan kernel 3 x 3
model.add(MaxPooling2D(pool_size=(2, 2))) #Variabel model di tambahkan dengan lib MaxPooling2D dengan ketentuan ukuran 2 x 2 pixcel
model.add(Flatten()) #Variabel model di tambahkan library Flatten
model.add(Dense(128, activation='tanh')) #Variabel model di tambahkan library Dense dengan fungsi tanh
model.add(Dropout(0.5)) #Variabel model di tambahkan library dropout untuk memangkas data tree sebesar 50 persen
model.add(Dense(num_classes, activation='softmax')) #Variabel model di tambahkan library Dense dengan data dari num_classes dan fungsi softmax
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy']) #Mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
print(model.summary()) #Mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
```

Listing 7.23 Kode program blok(14)

Hasil Skrinsut:

```
Model: "sequential_42"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_63 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_63 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_64 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_64 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
flatten_42 (Flatten)         (None, 1152)              0
_________________________________________________________________
dense_83 (Dense)             (None, 128)               147584
_________________________________________________________________
dropout_32 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_84 (Dense)             (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.287 Hasil Skrinsut

7.11.2.15 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

Penjelasan kode:

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

```
model.fit(np.concatenate((train_input, test_input)), #Melakukan training yang isi datanya dari join numpy menggunakan data train_input test_input
          np.concatenate((train_output, test_output)), #Kelanjutan data yang di gunakan pada join train_output test_output
          batch_size=32, epochs=10, verbose=2) #Menggunakan ukuran 32 bit dan epoch 10
```

Listing 7.24 Kode program blok(15)

Hasil Skrinsut:

```
Epoch 1/10
 - 38s - loss: 1.7842 - accuracy: 0.5857
Epoch 2/10
 - 39s - loss: 1.0848 - accuracy: 0.7044
Epoch 3/10
 - 38s - loss: 0.9728 - accuracy: 0.7290
Epoch 4/10
 - 38s - loss: 0.9141 - accuracy: 0.7411
Epoch 5/10
 - 38s - loss: 0.8725 - accuracy: 0.7492
Epoch 6/10
 - 38s - loss: 0.8460 - accuracy: 0.7551
Epoch 7/10
 - 38s - loss: 0.8250 - accuracy: 0.7597
Epoch 8/10
 - 38s - loss: 0.8032 - accuracy: 0.7651
Epoch 9/10
 - 37s - loss: 0.7919 - accuracy: 0.7654
Epoch 10/10
 - 38s - loss: 0.7786 - accuracy: 0.7686
<keras.callbacks.callbacks.History at 0x7fee61ec97b8>
```

Gambar 7.288 Hasil Skrinsut

7.11.2.16 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

Penjelasan kode:

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

Listing 7.25 Kode program blok(16)

Hasil Skrinsut:

```
[16] model.save("/content/mathsymbols.model") #Menyimpan model atau mengeksport model yang telah di jalan tadi
```

Gambar 7.289 Hasil Skrinsut

7.11.2.17 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

Penjelasan kode:

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

```
np.save('classes.npy', label_encoder.classes_) #Menyimpan label
    encoder dengan nama classes.npy
```

Listing 7.26 Kode program blok(17)

Hasil Skrinsut:

```
[17] np.save('/content/classes.npy', label_encoder.classes_) #Menyimpan label encoder dengan nama classes.npy
```

Gambar 7.290 Hasil Skrinsut

7.11.2.18 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

Penjelasan kode:

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

Listing 7.27 Kode program blok(18)

Hasil Skrinsut:

```
Model: "sequential_42"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_63 (Conv2D)           (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d_63 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_64 (Conv2D)           (None, 13, 13, 32)        9248
_________________________________________________________________
max_pooling2d_64 (MaxPooling (None, 6, 6, 32)          0
_________________________________________________________________
flatten_42 (Flatten)         (None, 1152)              0
_________________________________________________________________
dense_83 (Dense)             (None, 128)               147584
_________________________________________________________________
dropout_32 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_84 (Dense)             (None, 369)               47601
=================================================================
Total params: 205,329
Trainable params: 205,329
Non-trainable params: 0
_________________________________________________________________
None
```

Gambar 7.291 Hasil Skrinsut

7.11.2.19 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

Penjelasan kode:

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

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

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

Listing 7.28 Kode program blok(19)

Hasil Skrinsut:

Gambar 7.292 Hasil Skrinsut

7.11.2.20 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

Penjelasan kode:

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

```
predict("HASYv2/hasy-data/v2-00500.png") #Mencari prediksi
    menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2
    /hasy-data/v2-00500.png
predict("HASYv2/hasy-data/v2-00700.png") #Mencari prediksi
    menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2
    /hasy-data/v2-00700.png
```

Listing 7.29 Kode program blok(20)

Hasil Skrinsut:

```
Prediction: A, confidence: 0.74
Prediction: \pi, confidence: 0.48
Prediction: \alpha, confidence: 0.90
```

Gambar 7.293 Hasil Skrinsut

7.11.3 Penanganan Error

7.11.3.1 Terjadi error

1. terjadi error invalid argument, seperti pada gambar berikut:

```
OSError: [Errno 22] Invalid argument: 'HASYv2/hasy-data/v2-152218.png'
```

Gambar 7.294 terjadi Error 1

2. terjadi error Sequential has no attribute, seperti pada gambar berikut:

```
    super(TensorBoard, self).set_model(model)

  File "C:\ProgramData\Anaconda3\lib\site-packages\tensorflow_core\python\keras\callbacks.py", line
1532, in set_model
    self.log_dir, self.model._get_distribution_strategy())  # pylint: disable=protected-access

AttributeError: 'Sequential' object has no attribute '_get_distribution_strategy'
```

Gambar 7.295 terjadi Error 2

7.11.3.2 Solusi

1. solusi dari error 1 ialah: membenarkan argumen, karena terdapat typo

2. solusi dari error 2 ialah: proses running yang tidak sempurna pada blok sebelumnya, sehingga harus dijalan kan lagi dengan benar.

7.11.4 Bukti Tidak Plagiat

Gambar 7.296 Bukti tidak plagiat

7.11.5 Link Youtube

https://bit.ly/baktiListVideo

7.12 1174069 - Fanny Shafira Damayanti

7.12.1 Teori

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

Gambar 7.297 Illustrasi Tokenizer

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

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

```

Listing 7.30 K Fold Cross Validation

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

Gambar 7.298 Illustrasi K Fold Cross Validation

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

```
In [3]: import numpy as np
   ...: from sklearn.model_selection import train_test_split
   ...: x= np.arange(5).reshape((5, 2))
   ...: y = range(5)
   ...: print(x)
[[0 1]
 [2 3]
 [4 5]]

In [4]: list(y)
   ...: x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.33, random_state=42)
   ...: print(x_train)
   ...: print(x_test)
[[2 3]
 [4 5]]
[[0 1]]
```

Gambar 7.299 Illustrasi For train dan test in split

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

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

Gambar 7.300 Illustrasi penggunaan kolom Content

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

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

Gambar 7.301 Illustrasi fit tokenizer dan num_word=2000

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

Gambar 7.302 Illustrasi d train inputs = tokenizer.texts to matrix

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

7. Jelaskan apa maksud fungsi dari *d_train_outputs = np_utils.to_categorical(d['CLASS'].iloc*
   dan *d_test_outputs = np_utils.to_categorical(d['CLASS'].iloc[test_idx])* dalam kode program, dilengkapi dengan ilustrasi atau gambar.
   Maksud dari fungsi tersebut yaitu untuk merubah nilai vektor yang ada pada

atribut class menjadi bentuk matrix dengan pengurutan berdasarkan data index training dan testing.

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

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

```

Listing 7.31 Membuat model Neural Network

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

Gambar 7.303 Illustrasi d train inputs

Gambar 7.304 Illustrasi train outputs = np utils.to categorical

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

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

Listing 7.32 Compile model

model tersebut kemudian di compile atau di kembalikan kembali fungsi nilainya yangmana akan mengembalikan fungsi nilai loss nya berapa yang diambil dari fungsi adamax yang berberguna untuk mengetahui nilai lossnya kemudian metrics = acuracy merupakan akurasi dari nilai matrixnya. kemudian terdiri atas beberapa layer atau hiden layer. perbedaan antara deep learning dan DNN atau Deep Neural Network yaitu deep lerning merupakan pemakai algoritma dari DNN dan DNN merupakan algoritma yang ada pada deep learning.

Gambar 7.305 Illustrasi Neural Network

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

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

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

Gambar 7.306 Illustrasi perhitungan stride 1 max pooling

7.12.2 Praktek

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

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

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

```
# In[2]:load all images (as numpy arrays) and save their classes
#Menginisiasi variabel imgs dan classes dengan variabel array
    kosong
imgs = []
classes = []
#membuaka file hasy-data-labels.csv yang berada di folede HASYv2
    yang di inisialisasi menjadi csvfile
with open('F:// Semester 6/ Artificial Intelligence/Tugas 7/New
    Folder/hasy-data-labels.csv') as csvfile:
    #Menginisiasi variabel csvreader yang berisi method csv.
    reader yang membaca variabel csvfile
    csvreader = csv.reader(csvfile)
    # Menginisiasi variabel i dengan isi 0
    i = 0
    # membuat looping pada variabel csvreader
```

```
    for row in csvreader:
        # dengan ketentuan jika i lebihkecil daripada o
        if i > 0:
            # dibuat variabel img dengan isi keras untuk aktivasi neural network fungsi yang membaca data yang berada dalam folder HASYv2 dengan input nilai −1.0 dan 1.0
            img = keras.preprocessing.image.img_to_array(pil_image.open("F://Semester 6/Artificial Intelligence/Tugas 7/New Folder/" + row[0]))
            #Pembagian data yang ada pada fungsi img sebanyak 255.0
            img /= 255.0
            # Penambahan nilai baru pada imgs pada row ke 1 2 dan dilanjutkan dengan variabel img
            imgs.append((row[0], row[2], img))
            # Penambahan nilai pada row ke 2 pada variabel classes
            classes.append(row[2])
            # penambahan nilai satu pada variabel i
        i += 1
```

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

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

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

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

```
# Menginisiasi variabel test_output dengan np method asarray yang mana membuat array dengan isi row 1 dari data test
test_output = np.asarray(list(map(lambda row: row[1], test)))
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam',
              metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

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

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

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

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

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

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

```
# In[13]:try various model configurations and parameters to find the best
# Melakukan import library time
import time
#Menginisiasi variabel result dengan array kosong
results = []
# melakukan looping dengan ketentuan konvolusi 2 dimensi 1 2
for conv2d_count in [1, 2]:
    # menentukan ukuran besaran fixcel dari data atau konvert 1 fixcel mnjadi data yang berada pada codigan dibawah.
    for dense_size in [128, 256, 512, 1024, 2048]:
```

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

        start = time.time()
```

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

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

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

```
# mengkompile data model untuk mendapatkan data loss akurasi dan optimasi
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
# mencetak variabel model kemudian memunculkan kesimpulan berupa data total parameter, trainable paremeter dan bukan trainable parameter
print(model.summary())
```

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

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

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

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

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

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

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

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

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

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

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

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

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

```
# In[20]: grab an image (we'll just use a random training image for demonstration purposes)
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00010.png
predict("F://Semester 6/Artificial Intelligence/Tugas 7/New Folder/hasy-data/v2-00010.png")
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00500.png
predict("F://Semester 6/Artificial Intelligence/Tugas 7/New Folder/hasy-data/v2-00500.png")
# mencari prediksi menggunakan fungsi prediksi yang di buat tadi dari data di HASYv2/hasy-data/v2-00700.png
predict("F://Semester 6/Artificial Intelligence/Tugas 7/New Folder/hasy-data/v2-00700.png")
```

7.12.3 Penanganan Error

1. SS Error

Gambar 7.307 No Module Name error

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

7.12.4 Bukti Tidak Plagiat

Gambar 7.308 Tidak Melakukan Plagiat Pada Ch 7

7.13 1174086 - Tia Nur Candida

7.13.1 Teori

1. Jelaskan kenapa file teks harus di lakukan tokenizer. dilengkapi dengan ilustrasi atau gambar.
   file teks dilakukan tokenizer untuk diberikan urutan karakter dan unit dokumen yang ditentukan.

Gambar 7.309 Illustrasi Tokenizer

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

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

```

Listing 7.33 K Fold Cross Validation

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

Gambar 7.310 Illustrasi K Fold Cross Validation

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

Gambar 7.311 Illustrasi For train dan test in split

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

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

Gambar 7.312 Illustrasi penggunaan kolom Content

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

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

Gambar 7.313 Illustrasi fit tokenizer dan num_word=2000

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

Gambar 7.314 Illustrasi d train inputs = tokenizer.texts to matrix

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

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

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

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

```

Listing 7.34 Membuat model Neural Network

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

Gambar 7.315 Illustrasi d train inputs

Gambar 7.316 Illustrasi train outputs = np utils.to categorical

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

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

```

Listing 7.35 Compile model

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

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

Gambar 7.317 Illustrasi Neural Network

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

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

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

Gambar 7.318 Illustrasi perhitungan stride 1 max pooling

7.13.2 Praktek

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7.13.3 Penanganan Error

1. SS Error

```
  File "<ipython-input-14-d003036ef304>", line 1, in <module>
    model = Sequential()

NameError: name 'Sequential' is not defined
```

Gambar 7.319 No Module Name error

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

7.13.4 Bukti Tidak Plagiat

Gambar 7.320 Tidak Melakukan Plagiat Pada Ch 7

BAB 8

CHAPTER 8

8.1 1174083 - Bakti Qilan Mufid

Chapter 8 - Perkenalan Generative Adversarial Network

8.1.1 Teori

8.1.1.1 Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.

Menurut KBBI, Generator adalah pembangkit. Arti lainnya dari generator adalah alat pembangkit tenaga listrik. atau jika pada kecerdasan buatan bisa diartikan sebagai pemubuat. misalnya pembuat gambar, pembuat musik, dll. jika perumpamaan mahasiswa sebagai generator maka dapat digambarkan seperti berikut:

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

Gambar 8.1 gambaran penjelasan no. 1

8.1.1.2 Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatornya.

arti kata diskriminator adalah rangkaian dalam berbagai alat. berasal dari kata diskriminasi yang merujuk pada pelayanan yang tidak adil terhadap individu tertentu, di mana layanan ini dibuat berdasarkan karakteristik yang diwakili oleh individu tersebut. dalam machine learning, Discrimination itu bisa ada dua, yaitu

1. Disparate Treatment(perawatan berbeda) atau membedakan/ mengklasifikasikan sesuatu atau seseorang

2. Disparate Impact (dampak berbeda) ialah melihat konsekuensi klasifikasi/ pengambilan keputusan pada kelompok tertentu.

berikut ilustrasi jika dosen sebagai diskriminator:

Gambar 8.2 gambaran penjelasan no. 2

8.1.1.3 Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur generator dibuat

aristektur generator didalam GAN terdiri dari lima lapisan, yaitu: satu input layer, tiga dense layer dan satu output layer. bisa dilihat arsitektur geerator pada GAN-Project seperti gambar berikut:

13970549349376		
dense_1:Dense	Input:	(None, None, 100)
	Output:	(None, None, 500)
dense_2:Dense	Input:	(None, None, 500)
	Output:	(None, None, 500)
dense_3:Dense	Input:	(None, None, 500)
	Output:	(None, None, 784)
reshape_1:Reshape	Input:	(None, None, 784)
	Output:	(None, 28, 28)

Gambar 8.3 gambar arsitektur generator pada GAN-Project

untuk ilustrasinya seperti berikut:

Gambar 8.4 gambaran penjelasan no. 3

8.1.1.4 Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminator dibuat aristektur diskriminator didalam GAN terdiri dari lima lapisan, yaitu: satu input layer, tiga dense layer dan satu output layer. bisa dilihat arsitektur diskriminator pada GAN-Project seperti gambar berikut:

Gambar 8.5 gambar arsitektur generator pada GAN-Project

untuk ilustrasinya seperti berikut:

Gambar 8.6 gambaran penjelasan no. 4

8.1.1.5 Jelaskan dengan ilustrasi gambar apa itu latent space.

latent space adalah space atau spot yang tersembunyi, sehingga akan mengambil titik data yang dekat dan yang mirip. ilustrasinya seperti pada gambar berikut:

Gambar 8.7 gambaran penjelasan no. 5

8.1.1.6 Jelaskan dengan ilustrasi gambar apa itu adversarial play

Dalam GAN adversial play ialah persaingan antara generator dan diskriminator. bisa dilihat pada ilustrasi gambar berikut:

Gambar 8.8 gambaran penjelasan no. 6

8.1.1.7 Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium

nash equilibrium adalah strategi yang dipilih oleh masing-masing pemain, dengan strategi pemain lain tertentu. Strategi ini merupakan strategi terbaik dari masing-masing pemain dengan mempertimbangkan strategi tertentu yang diambil oleh pemain lainnya, atau yang disebut dengan Best Respons.

SAM \ TOM	A	B
A	1,1	1,-1
B	-1,1	0,0

Gambar 8.9 gambaran penjelasan no. 7

8.1.1.8 Sebutkan dan jelaskan contoh-contoh implementasi dari GAN

Contoh implementasi dari GAN salah satunya adalah meng-generate wajah kucing, seperti pada gambar berikut:

Gambar 8.10 gambaran penjelasan no. 8

8.1.1.9 Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer)

untuk gambar arsitektur nya ada pada gambar no. 3 sedangkan untuk kode programnya seperti berikut:

```
def generator_model():
    model = Sequential([
        Dense(1024, input_dim=100, activation='tanh'),
        Dense(128*7*7),
        Reshape((7, 7, 128)),
        UpSampling2D(size=(2, 2)),
        Conv2D(64, (5, 5), padding='same', activation='tanh'),
        UpSampling2D(size=(2, 2)),
        Conv2D(1, (5, 5), padding='same', activation='tanh')
    ])
    return model

generator_model().summary()
```

Listing 8.1 Kode program arsitektur generator

8.1.1.10 Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuath input layer, 3 buah hidden layer, dan satu output layer.

untuk gambar arsitektur nya ada pada gambar no. 4 sedangkan untuk kode programnya seperti berikut:

```
def discriminator_model():
    model = Sequential([
        Conv2D(64, (5, 5), padding='same', input_shape=(28, 28, 1),
    activation='tanh'),
        MaxPooling2D(pool_size=(2, 2)),
        Conv2D(128, (5, 5),activation='tanh'),
        MaxPooling2D(pool_size=(2, 2)),
        Flatten(),
        Dense(1024, activation='tanh'),
        Dense(1, activation='sigmoid')
    ])
    return model

discriminator_model().summary()
```

Listing 8.2 Kode program arsitektur diskriminator

8.1.1.11 Jelaskan bagaimana kaitan output dan input antara generator dan diskriminator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

kaitan antara output pada generator dan inputan pada diskriminator adalah suatu keterhubungan yang akan menghasilkan data yang benar, karena output dari generator metupakan inoutan pada diskriminator, berikut gambarannya:

Gambar 8.11 gambaran penjelasan no. 9

8.1.1.12 Jelaskan apa perbedaan antara Kullback-Leibler divergence (KL divergence)/relative entropy, Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to the average dalam mengukur kualitas dari model.

diantara keduanya yang paling terkenal adalah KL divergence, karena KL divergence memiliki beberaa sifat/rumus yang bagus .salah satunya adalah KL[q; p] jenis daerah yang di mana q(x) memiliki massa non-null dan p(x) memiliki massa null. Ini mungkin terlihat seperti bug, tetapi sebenarnya ini merupakan fitur dalam situasi tertentu. sedangkan JS tidak memilikinya,

8.1.1.13 Jelaskan apa itu fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

fungsi objektif merupakan fungsi yang akan mengambil parameter data dan model sebagai argumen, dan dapat dievaluasi untuk mengembalikan angka sehingga mampu membedakan gambar yang asli dan yang di buat/fake.

8.1.1.14 Jelaskan apa itu scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

Inception Score digunakan untuk mengukur seberapa realistis output dari GAN, dimana ada dua parameter, yaitu : gambarnya punya variasi dan setiap gambar jelas terlihat seperti sesuatu. Frechet Inception Distance adalah ukuran kesamaan antara dua dataset gambar. Itu terbukti berkorelasi baik dengan penilaian manusia terhadap kualitas visual dan paling sering digunakan untuk mengevaluasi kualitas sampel Generative Adversarial Networks. FID dihitung dengan menghitung jarak Frchet antara dua Gaussians dipasang ke representasi fitur dari jaringan Inception.

8.1.1.15 Jelaskan kelebihan dan kekurangan GAN

- Kelebihan
  1. GAN Menghasilkan data baru yang bisa hampir mirip dengan data asli. Karena hasil pelatihannya, GAN dapat menghasilkan data gambar, teks, audio, dan video yang dapat dibilang hampir mirip dengan yang aslinya. Berkat hal tersebut, GAN dapat digunakan dalam sistem marketing, e-commerce, games,iklan, dan industri lainnya
  2. GAN mempelajari representasi data secara internal sehingga beberapa masalah pada machine learning dapat diatasi dengan mudah
  3. Discriminator yang sudah dilatih dapat menjadi sebuah classifier atau pendeteksi jika data sudah sesuai. Karena Discriminator yang akan menjadi tidak efisien berkat seringnya dilatih
  4. GAN dapat dilatih menggunakan data yang belum dilabeled
- Kekurangan
  1. Data saat diproses oleh metode gan tidak konvergensi
  2. Jenis sampel yang dihasilkan oleh generator terbatas karena modenya terbatas
  3. Ketidak seimbangnya antara generator dan discriminator dapat menyebabkan overfitting atau terlalu dekat dengan hasil sampel
  4. Sangat sensitif dengan data yang sudah diinisiasi sebelumnya

8.1.2 Praktek

8.1.2.1 Jelaskan apa itu 3D convolutions

3D convolutions merupakan operasi konvolusi 3D yang menerapkan filter 3D ke data

input dengan tiga arah, yaitu x,y, dan z. fitur ini menciptakan sebuah daftar peta yang ditumpuk. agar lebih mudah saya menggunakan sebuah ilustrasi seperti berikut:

Gambar 8.12 gambaran 3D Convolutions

8.1.2.2 Jelaskan dengan kode program arsitektur dari generator networknya, beserta penjelasan input dan output dari generator network.

```
def build_generator():#nama kelasnya
    """
    Create a Generator Model with hyperparameters values defined as
    follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5

    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model
```

kode diatas merupakan generator network yang terdiri dari lima layer, satu input layer, tiga hidden layer, dan satu output layer.Dan prosesnya dapat digambarkan seperti berikut:

Gambar 8.13 gambaran proses generator network

penjelasan

- Input layer mengambill 100-dimensi sampel dari distribusi Gaussian(normal) dan meneruskan tensor ke. hidden layer pertama tanpa ada modifikasi
- ketiga hidden layer adalah dense layer dengan unit masing-masing 500, 500, dan 784. dimana dense layer pertama mengkonversi bentuk tensor (batch_size, 100) ke bentuk tensor(batch_size, 500). sedangkan pada layer dense kedua menghasilkan bentuk tensor (batch_size, 500), dan layer dense ketiga menghasilkan (batch_size, 784)
- Output layer, tensor akan dibentuk kembali dari bentuk tensor (batch_size, 784) menjadi (batch_size, 28, 28) artinya hasil dari generator ini akan menghasilkan banyak gambar, dimana setiap gambarnya memiliki ukuran (28, 28).

8.1.2.3 Jelaskan dengan kode program arsitektur dari diskriminator network, beserta penjelasan input dan outputnya.

```
def build_discriminator():
    """
    Create a Discriminator Model using hyperparameters values defined
     as follows
    """
    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
```

```
                strides=dis_strides[0],
                padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)

    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model
```

kode diatas merupakan diskriminator network, dimana prosesnya dapat digambarkan seperti berikut:

Gambar 8.14 gambaran proses diskriminator network

penjelasan

- awalnya diskriminator menerima input dengan bentuk 28x28.
- Input layer mengambil tensor input dan meneruskannya ke hidden layer pertama tanpa modifikasi apapun.
- lalu flattens layer akan meratakan tensor menjadi 784-dimensi vektor, yang akan diteruskan ke hidden layer (dense layer) pertama. hidden layer pertama dan kedua aka memodifikasinya menjadi vektor 500-dimensi.
- Output layer mesuk kedalam dense layer, dengan satu unit neuron dan sigmoid sebagai fungsi aktivasi. sigmoid akan menghasilkan nilai tunggal, 0 atau 1. Nilai 0 akan menunjukan bahwa gambar yang diberikan palsu. sebaliknya jika nilai 1 maka gambar asli.

8.1.2.4 Jelaskan proses training 3D-GANs

Proses training 3D-GANs dilakukan seperti langkah-langkah berikut:

- Terdapat sebuah vektor noise dengan dimensi 200 dari distribusi Gaussian(normal).
- Meng-generate gambar palsu menggunakan model generator.
- Melatih jaringan generator dengan gambar yang asli(sampel dari data yang real) dan dengan gambar palsu yang dihasilkan oleh generator.
- Gunakan adversial model untuk melatih generator model, jangan melatih diskriminator model.
- Ulangi langkah ini dengan jumlah epoch tertentu.

8.1.2.5 Jelaskan bagaimana melakukan settingan awal chapter 02 untuk memenuhi semua kebutuhan sebelum melanjutkan ke tahapan persiapan data.

Sebelum melakukan tahapan persiapan data, kita harus melakukan beberapa hal seperti berikut:

- membaca buku panduan atau keterangan dari Generative-Adversial-Network project ini. bisa didowload di portal kampus keren atau di github https://github.com/PacktP Adversarial-Networks-Projects
- persiapkan laptop/pc dengan spek yang cukup bagus(tidak kentang). jika tidak ada, bisa menggunakan google colab(cara penggunaanya akan dijelaskan di video).
- usahakn versi yang terinstall sama dengan versi yang ada pada requirement.txt agar terhindar dari error.

8.1.2.6 Jelaskan tentang dataset yang digunakan, dari mulai tempat unduh, cara membuka dan melihat data. sampai deskripsi dari isi dataset dengan detail penjelasan setiap folder/file yang membuat orang awam paham.

Penjelasan dataset

- dataset bisa didownload pada halaman: http://3dshapenets.cs.princeton.edu/3DShapeNets
- Untuk membuka data, setelah didownload cukup di eksrak saja menggunakan winrar/ 7zip
- Untuk mengetahui lebih jelasnya tentang dataset ini bisa dibuka saja file README.md, disana akan dijelaskan semuanya.

Gambar 8.15 Isi dari dataser.zip

8.1.2.7 Jelaskan apa itu voxel dengan ilustrasi dan bahasa paling awam

Secara singkat, voxel dapat diartikan sebagai 3D pixel atau pixel dalam bentuk 3D. seperti pada ilustrasi berikut:

Gambar 8.16 Pixel VS Voxel

8.1.2.8 Visualisasikan dataset tersebut dalam tampilan visual plot, jelaskan cara melakukan visualisasinya

```
import scipy.io as io
```

```
import matplotlib.pyplot as plt
import scipy.ndimage as nd
import numpy as np
from mpl_toolkits.mplot3d import Axes3D

voxels = io.loadmat("/content/3DShapeNets/volumetric_data/flower_pot
    /30/test/flower_pot_000000010_1.mat")['instance']
voxels = np.pad(voxels, (1, 1), 'constant', constant_values=(0, 0))
voxels = nd.zoom(voxels, (2, 2, 2), mode='constant', order=0)

fig = plt.figure()
ax = Axes3D(fig)
ax.voxels(voxels, edgecolor="chartreuse")

plt.show()
plt.savefig('flower_pot')
```

Kode diatas berfungsi untuk melakukan visualisasi dataset dalam tampilan plot. dengan tahapan import library, load data file .mat, dan lakukan read memakai matplotlib. dan hasilnya seperti berikut

Gambar 8.17 Visualisasi dataset(contoh)

8.1.2.9 Buka file run.py jelaskan perbaris kode pada fungsi untuk membuat generator yaitu build generator

```
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5
```

potongan kode diatas berfungsi sebagai penentuan nilai untuk hyperparameters yang berbeda.

```
input_layer = Input(shape=gen_input_shape)
```

potongan kode diatas berfungsi untuk membuat input layer

```
# First 3D transpose convolution(or 3D deconvolution) block
a = Deconv3D(filters=gen_filters[0],
             kernel_size=gen_kernel_sizes[0],
             strides=gen_strides[0])(input_layer)
a = BatchNormalization()(a, training=True)
a = Activation(activation='relu')(a)
```

potongan kode diatas berfungsi untuk menambahkan 3D transpose pertama.

```
# Next 4 3D transpose convolution(or 3D deconvolution) blocks
for i in range(gen_convolutional_blocks - 1):
    a = Deconv3D(filters=gen_filters[i + 1],
                 kernel_size=gen_kernel_sizes[i + 1],
                 strides=gen_strides[i + 1], padding='same')(a)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation=gen_activations[i + 1])(a)
```

potongan kode diatas berfungsi untuk menambahkan empat 3D transpose lainya.

```
gen_model = Model(inputs=[input_layer], outputs=[a])
```

potongan kode diatas berfungsi untuk membuat model keras dan menentukan input dan output untuk network generator.

8.1.2.10 jelaskan juga fungsi untuk membangun diskriminator pada fungsi build discriminator.

```
dis_input_shape = (64, 64, 64, 1)
dis_filters = [64, 128, 256, 512, 1]
dis_kernel_sizes = [4, 4, 4, 4, 4]
dis_strides = [2, 2, 2, 2, 1]
dis_paddings = ['same', 'same', 'same', 'same', 'valid']
dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
```

potongan kode diatas berfungsi sebagai penentuan nilai untuk hyperparameters yang berbeda.

```
dis_input_layer = Input(shape=dis_input_shape)
```

potongan kode diatas berfungsi untuk membuat input layer, berupa gambar 3D dengan dimensi 64x64x64x1

```
# The first 3D Convolutional block
a = Conv3D(filters=dis_filters[0],
           kernel_size=dis_kernel_sizes[0],
           strides=dis_strides[0],
           padding=dis_paddings[0])(dis_input_layer)
# a = BatchNormalization()(a, training=True)
a = LeakyReLU(dis_alphas[0])(a)
```

potongan kode diatas berfungsi untuk menambahkan 3D transpose pertama.

```
    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)
```

potongan kode diatas berfungsi untuk menambahkan empat 3D transpose lainya.

```
    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
```

potongan kode diatas berfungsi untuk membuat model keras dan menentukan input dan output untuk network generator.

8.1.2.11 jelaskan apa maksud dari kode program name == ' main '

```
if __name__ == '__main__':
```

Listing 8.3 Kode program run.py

maksudnya kode tersebut memiliki fungsi jika interpreter python menjalankan kode tersebut, maka akan menetapkan variable name untuk memiliki nilai main. jika file ini di import dari modul lain, maka name akan ditetapkan ke nama modul tersbut.

8.1.2.12 jelaskan secara detil perbaris dan per parameter apa arti dari kode program :

Penjelasan kode:

```
# In[soal 12]
    object_name = "airplane" #membuat variabel
    data_dir = "data/3DShapeNets/volumetric_data/" \
               "{}/30/train/*.mat".format(object_name) #menunjukan
    direktori dari variable sebelumnya
    gen_learning_rate = 0.0025 #membuat rate pada generator
    dis_learning_rate = 10e-5 #membuat rate pada diskriminator
    beta = 0.5 #
    batch_size = 1 #menentukan ukuran kumpulan gambar
    z_size = 200 #
    epochs = 1 #menetukkan masa periode. setiap epoch memiliki semua
    batch
    MODE = "train" #membuat traon mode
```

Listing 8.4 Kode program run.py

8.1.2.13 Jelaskan secara detil dari kode program pembuatan dan kompilasi arsitektur berikut :

```
    gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
    dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)

    discriminator = build_discriminator()
    discriminator.compile(loss='binary_crossentropy', optimizer=
    dis_optimizer)

    generator = build_generator()
    generator.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

Listing 8.5 Kode program run.py

penjelasanya ialah membuat model dari generator model dan diskriminator model,

8.1.2.14 Jelaskan secara detil kode program untuk membuat dan melakukan kompilasi model adversarial berikut:

```
# In[soal 14]
    discriminator.trainable = False

    input_layer = Input(shape=(1, 1, 1, z_size))
    generated_volumes = generator(input_layer)
    validity = discriminator(generated_volumes)
    adversarial_model = Model(inputs=[input_layer], outputs=[validity
    ])
    adversarial_model.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

Listing 8.6 Kode program run.py

penjelasanya ialah membuat model diskriminator tidak di training dan membuat model adversarial.

8.1.2.15 Jelaskan Ekstrak dan load data kursi dengan menggunakan fungsi getVoxelsFormat dan get3DImages yang digunakan pada kode program berikut :

```
# In[soal 15]
    print("Loading data...")
    volumes = get3DImages(data_dir=data_dir)
    volumes = volumes[..., np.newaxis].astype(np.float)
    print("Data loaded...")
```

Listing 8.7 Kode program run.py

penjelasanya ialah melakukan print loading data, membuat variabel volumes untuk mendapatkan data gamabr 3D.

8.1.2.16 Jelaskan maksud dari kode program instansiasi TensorBoard yang menambahkan generator dan diskriminator pada program berikut:

```
# In[soal 16]
    tensorboard = TensorBoard(log_dir="logs/{}".format(time.time()))
    tensorboard.set_model(generator)
    tensorboard.set_model(discriminator)
```

Listing 8.8 Kode program run.py

penjelasanya ialah membuat variabel tensorboard untuk melakukan pencatatan log, dan men-set model generator dan model diskriminator

8.1.2.17 Jelaskan apa fungsi dari np reshape ones zeros pada kode program berikut dengan parameternya:

Penjelasan kode:

```
# In[soal 17]
    labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1, 1, 1)
    )
    labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1, 1,
    1))
```

Listing 8.9 Kode program run.py

membuat label_real untuk menetapkan bahwa jika output 1 itu menunjukan gambar real, dan label_fake untuk menunjukan bahwa output 0 untuk data fake/palsu.

8.1.2.18 Jelaskan kenapa harus ada perulangan dalam meraih epoch. Dan jelaskan apa itu epoch terkait kode program berikut:

```
# In[soal 18]
    if MODE == 'train':
        for epoch in range(epochs):
            print("Epoch:", epoch)

            gen_losses = []
            dis_losses = []
```

Listing 8.10 Kode program run.py

melakukan perulangan jika MODE = train maka akan diulangi terus menerus sampai epoch(periode) terpenuhi dengan disertai keterangan generator dan diskriminator losses.

8.1.2.19 Jelaskan apa itu batches dan kaitannya dengan kode program berikut, dan kenapa berada di dalam epoch:

```
# In[soal 19]
            number_of_batches = int(volumes.shape[0] / batch_size)
            print("Number of batches:", number_of_batches)
            for index in range(number_of_batches):
                print("Batch:", index + 1)
```

Listing 8.11 Kode program run.py

menghitung jumlah batch yang telah ditrain

8.1.2.20 Berikut adalah kode program pengambilan gambar dan noise. Jelaskan apa fungsi np.random.normal serta astype, serta jelaskan apa arti parameter titik dua dan jelaskan isi dari z sample dan volumes batch:

```
# In[soal 20]
                z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]
```

Listing 8.12 Kode program run.py

untuk membersihkan gambar dari noise dan menyesuaikan shape.

8.1.2.21 Berikut adalah kode program generator gambar palsu. Jelaskan apa fungsi generator.predict on batch, serta jelaskan apa arti parameter z sample:

```
# In[soal 21]
                # Next, generate volumes using the generate network
                gen_volumes = generator.predict_on_batch(z_sample)
```

Listing 8.13 Kode program run.py

untuk meng-generate volume menggunakan model jaringan generator.

8.1.2.22 Berikut adalah kode program training diskriminator dengan gambar palsu dari generator dan gambar asil. Jelaskan apa maksudnya harus dilakukan training diskriminator secara demikian dan jelaskan apa isi loss fake dan loss real serta d loss dan fungsi train on batch.

```
# In[soal 22]
                discriminator.trainable = True
                if index % 2 == 0:
                    loss_real = discriminator.train_on_batch(volumes_batch, labels_real)
                    loss_fake = discriminator.train_on_batch(gen_volumes, labels_fake)

                    d_loss = 0.5 * np.add(loss_real, loss_fake)
                    print("d_loss:{}".format(d_loss))

                else:
                    d_loss = 0.0
```

Listing 8.14 Kode program run.py

untuk membuat diskriminator bisa meload gambar fake dan real dari model generator. dengan disertai genrator dan diskriminator loss, agar terliaht seberapa baik kualitas yang dihasilkan.

8.1.2.23 Berikut adalah kode program training model adversarial yang terdapat generator dan diskriminator. Jelaskan apa bagaimana proses terbentuknya

parameter z dan g loss:

```
# In[soal 23]
                discriminator.trainable = False
                """
                Train the generator network
                """
                z = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                g_loss = adversarial_model.train_on_batch(z, labels_real)
                print("g_loss:{}".format(g_loss))

                gen_losses.append(g_loss)
                dis_losses.append(d_loss)
```

Listing 8.15 Kode program run.py

untuk mentrain model generator dan variabel g_loss untuk melakukan perbandingan antara label gambar yang asli.

8.1.2.24 Berikut adalah kode program generate dan menyimpan gambar 3D setelah beberapa saat setiap epoch. Jelaskan mengapa ada perulangan dengan parameter tersebut, serta jelaskan arti setiap variabel beserta perlihatkan isinya dan artikan isinya :

```
# In[soal 24]
                # Every 10th mini-batch, generate volumes and save them
                if index % 10 == 0:
                    z_sample2 = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                    generated_volumes = generator.predict(z_sample2, verbose=3)
                    for i, generated_volume in enumerate(generated_volumes[:5]):
                        voxels = np.squeeze(generated_volume)
                        voxels[voxels < 0.5] = 0.
                        voxels[voxels >= 0.5] = 1.
                        saveFromVoxels(voxels, "results/img_{}_{}_{}".format(epoch, index, i))
```

Listing 8.16 Kode program run.py

untuk melakukan perulangan dimana setiap terdapat 10 mini-batch akan meng-generate volumenya dan akan menyimpannya.

8.1.2.25 Berikut adalah kode program menyimpan average losses setiap epoch. Jelaskan apa itu tensorboard dan setiap parameter yang digunakan pada kode program ini :

```
# In[soal 25]
```

```
            # Write losses to Tensorboard
            write_log(tensorboard, 'g_loss', np.mean(gen_losses), epoch)
            write_log(tensorboard, 'd_loss', np.mean(dis_losses), epoch)
```

Listing 8.17 Kode program run.py

untuk menuliskan log losses ke Tensorboard

8.1.2.26 Berikut adalah kode program menyimpan model. Jelaskan apa itu format h5 dan penjelasan dari kode program berikut :

```
# In[soal 26]
        generator.save_weights(os.path.join("models", "generator_weights.h5"))
        discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))
```

Listing 8.18 Kode program run.py

untuk menyimpan berat model generator dan model diskriminator kedalam file h5.

H5 adalah Hierarchical Data Format 5 File. File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah. File H5 biasanya digunakan di luar angkasa, fisika, teknik, keuangan, penelitian akademis, genomik, astronomi, instrumen elektronik, dan bidang medis.

8.1.2.27 Berikut adalah kode program testing model. Jelaskan dengan ilustrasi gambar dari mulai meload hingga membuat gambar 3D dengan menggunakan z sample, bisakah parameter z sample tersebut diubah2? :

```
        generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
        discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)

        # Generate 3D models
        z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
        generated_volumes = generator.predict(z_sample, verbose=3)

        for i, generated_volume in enumerate(generated_volumes[:2]):
            voxels = np.squeeze(generated_volume)
            voxels[voxels < 0.5] = 0.
            voxels[voxels >= 0.5] = 1.
            saveFromVoxels(voxels, "results/gen_{}".format(i))
```

Listing 8.19 Kode program run.py

untuk membuat mode predict/prediksi. dimana dia melakukan pebuatan model generator, model diskriminator dan meload data h5 yang sudah dibuat tadi dan menggenerate nya kedalam model 3D, lalu menyimpan model voxel nya ke folde results.

8.1.3 Penanganan Error

8.1.3.1 Terjadi error

1. terjadi error Summary has no attribute, seperti pada gambar berikut:

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-4-d9ec96389e6a> in <module>()
    233
    234             # Write losses to Tensorboard
--> 235             write_log(tensorboard, 'g_loss', np.mean(gen_losses), epoch)
    236             write_log(tensorboard, 'd_loss', np.mean(dis_losses), epoch)
    237

<ipython-input-4-d9ec96389e6a> in write_log(callback, name, value, batch_no)
     93
     94 def write_log(callback, name, value, batch_no):
---> 95     summary = tf.Summary()
     96     summary_value = summary.value.add()
     97     summary_value.simple_value = value

AttributeError: module 'tensorflow' has no attribute 'Summary'
```

Gambar 8.18 terjadi Error 1

8.1.3.2 Solusi

1. solusi dari error 1 ialah: dengan menginstall versi tensorflow 1.xx

```
(base) C:\Users\Bakti Qilan>pip install tensorflow==1.2
```

Gambar 8.19 solusi Error 1

8.1.4 Bukti Tidak Plagiat

Gambar 8.20 Bukti tidak plagiat

8.1.5 Link Youtube

https://bit.ly/baktiListVideo

8.2 1174066 - D.Irga B. Naufal Fakhri

8.2.1 Teori

8.2.1.1 Apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.

Generator adalah pembuat dan didalam program, generator adalah pembuat sesuatu. misalnya:

Gambar 8.21 Penjelasan no.1

8.2.1.2 Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatornya.

Arti kata diskriminator adalah rangkaian dalam berbagai alat. berasal dari kata diskriminasi yang merujuk pada pelayanan yang tidak adil terhadap individu tertentu, di mana layanan ini dibuat berdasarkan karakteristik yang diwakili oleh individu tersebut. dalam machine learning, Discrimination itu bisa ada dua, yaitu:

1. Disparate Treatment(perawatan berbeda) atau membedakan/ mengklasifikasikan sesuatu atau seseorang
2. Disparate Impact (dampak berbeda) ialah melihat konsekuensi klasifikasi/ pengambilan keputusan pada kelompok tertentu.

Gambar 8.22 Penjelasan no.2

8.2.1.3 Bagaimana arsitektur generator dibuat

Aristektur generator didalam GAN terdiri dari lima lapisan, yaitu: satu input layer, tiga dense layer dan satu output layer.

Gambar 8.23 Arsitektur generator pada GAN-Project

Dengan ilustrasi sebagai berikut:

Gambar 8.24 Penjelasan no.3

8.2.1.4 Bagaimana arsitektur diskriminator dibuat

aristektur diskriminator didalam GAN terdiri dari lima lapisan, yaitu: satu input layer, tiga dense layer dan satu output layer. Diskriminator pada GAN-Project:

Gambar 8.25 Arsitektur Diskriminator pada GAN-Project

Dengan ilustrasi sebagai berikut:

Gambar 8.26 Penjelasan no.4

8.2.1.5 Apa itu latent space.

Latent space adalah space atau spot yang tersembunyi, sehingga akan mengambil titik data yang dekat dan yang mirip. Dengan ilustrasi sebagai berikut:

Gambar 8.27 Penjelasan no.5

8.2.1.6 Apa itu adversarial play

Dalam GAN adversial play ialah persaingan antara generator dan diskriminator.

Gambar 8.28 Penjelasan no.6

8.2.1.7 Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium

Nash equilibrium adalah strategi yang dipilih oleh masing-masing pemain, dengan strategi pemain lain tertentu. Strategi ini merupakan strategi terbaik dari masing-masing pemain dengan mempertimbangkan strategi tertentu yang diambil oleh pemain lainnya, atau yang disebut dengan Best Respons.

		TOM	
		A	B
SAM	A	1,1	1,-1
	B	-1,1	0,0

Gambar 8.29 Penjelasan no.7

8.2.1.8 Sebutkan dan jelaskan contoh-contoh implementasi dari GAN

Contoh implementasi dari GAN salah satunya adalah men-generate wajah kucing

Gambar 8.30 Penjelasan no.8

8.2.1.9 Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer)

Gambar 8.31 Arsitektur generator pada GAN-Project

```
def generator_model():
    model = Sequential([
        Dense(1024, input_dim=100, activation='tanh'),
        Dense(128*7*7),
        Reshape((7, 7, 128)),
        UpSampling2D(size=(2, 2)),
        Conv2D(64, (5, 5), padding='same', activation='tanh'),
        UpSampling2D(size=(2, 2)),
        Conv2D(1, (5, 5), padding='same', activation='tanh')
    ])
    return model

generator_model().summary()
```

Listing 8.20 Kode program arsitektur generator

8.2.1.10 Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuath input layer, 3 buah hidden layer, dan satu output layer.

Gambar 8.32 Arsitektur Diskriminator pada GAN-Project

```
def discriminator_model():
    model = Sequential([
        Conv2D(64, (5, 5), padding='same', input_shape=(28, 28, 1),
    activation='tanh'),
        MaxPooling2D(pool_size=(2, 2)),
        Conv2D(128, (5, 5),activation='tanh'),
        MaxPooling2D(pool_size=(2, 2)),
        Flatten(),
        Dense(1024, activation='tanh'),
        Dense(1, activation='sigmoid')
```

```
    ])
    return model

discriminator_model().summary()
```

Listing 8.21 Kode program arsitektur diskriminator

8.2.1.11 Kaitan output dan input antara generator dan diskriminator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

Kaitan antara output pada generator dan inputan pada diskriminator adalah suatu keterhubungan yang akan menghasilkan data yang benar, karena output dari generator merupakan inoutan pada diskriminator

Gambar 8.33 Penjelasan no.9

8.2.1.12 Perbedaan antara Kullback-Leibler divergence (KL divergence)/relative entropy, Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to the average dalam mengukur kualitas dari model.

diantara keduanya yang paling terkenal adalah KL divergence, karena KL divergence memiliki beberaa sifat/rumus yang bagus .salah satunya adalah KL[q; p] jenis daerah yang di mana q(x) memiliki massa non-null dan p(x) memiliki massa null. Ini mungkin terlihat seperti bug, tetapi sebenarnya ini merupakan fitur dalam situasi tertentu. sedangkan JS tidak memilikinya,

8.2.1.13 Fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

fungsi objektif merupakan fungsi yang akan mengambil parameter data dan model sebagai argumen, dan dapat dievaluasi untuk mengembalikan angka sehingga mampu membedakan gambar yang asli dan yang di buat/fake.

8.2.1.14 Scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

Inception Score digunakan untuk mengukur seberapa realistis output dari GAN, dimana ada dua parameter, yaitu: gambarnya punya variasi dan setiap gambar jelas terlihat seperti sesuatu. Frechet Inception Distance adalah ukuran kesamaan antara dua dataset gambar. Itu terbukti berkorelasi baik dengan penilaian manusia terhadap kualitas visual dan paling sering digunakan untuk mengevaluasi kualitas sampel Generative Adversarial Networks. FID dihitung dengan menghitung jarak Frchet antara dua Gaussians dipasang ke representasi fitur dari jaringan Inception.

8.2.1.15 Kelebihan dan kekurangan GAN

- Kelebihan
  1. GAN Menghasilkan data baru yang bisa hampir mirip dengan data asli. Karena hasil pelatihannya, GAN dapat menghasilkan data gambar, teks, audio, dan video yang dapat dibilang hampir mirip dengan yang aslinya. Berkat hal tersebut, GAN dapat digunakan dalam sistem marketing, e-commerce, games,iklan, dan industri lainnya
  2. GAN mempelajari representasi data secara internal sehingga beberapa masalah pada machine learning dapat diatasi dengan mudah
  3. Discriminator yang sudah dilatih dapat menjadi sebuah classifier atau pendeteksi jika data sudah sesuai. Karena Discriminator yang akan menjadi tidak efisien berkat seringnya dilatih
  4. GAN dapat dilatih menggunakan data yang belum dilabeled
- Kekurangan
  1. Data saat diproses oleh metode gan tidak konvergensi
  2. Jenis sampel yang dihasilkan oleh generator terbatas karena modenya terbatas
  3. Ketidak seimbangnya antara generator dan discriminator dapat menyebabkan overfitting atau terlalu dekat dengan hasil sampel
  4. Sangat sensitif dengan data yang sudah diinisiasi sebelumnya

8.2.2 Praktek

8.2.2.1 Jelaskan apa itu 3D convolutions

3D convolutions adalah operasi konvolusi 3D yang menerapkan filter 3D ke data input dengan tiga arah, yaitu x,y, dan z. fitur ini menciptakan sebuah daftar peta yang ditumpuk.

Gambar 8.34 Penjelasan praktek no.1

8.2.2.2 Jelaskan dengan kode program arsitektur dari generator networknya, beserta penjelasan input dan output dari generator network.

```
def build_generator():
    """
    Create a Generator Model with hyperparameters values defined as
    follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5

    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model
```

Kode di atas akan melakukan create generator ialah gloss, Bentuk jaringan Generator dapat dilihat berkebalikan dengan struktur jaringan saraf pada umumnya. Generator biasanya menerima input sebuah vektor z, yang kemudian mengubahnya menjadi sebuah output 3D atau 3 dimensi.

Gambar 8.35 gambaran proses generator network

8.2.2.3 Jelaskan dengan kode program arsitektur dari diskriminator network, beserta penjelasan input dan outputnya.

```
def build_discriminator():
    """
    Create a Discriminator Model using hyperparameters values defined
     as follows
    """

    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
               strides=dis_strides[0],
               padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)
```

```
    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model
```

Diskrimanator adalah d_loss, Jaringan Discriminator merupakan jaringan klasifikasi biner yang menerima input gambar tiga dimensi dan mengeluarkan klasifikasi menyatakan input gambar adalah gambar asli dari dataset atau merupakan gambar buatan Generator. Diskriminator dilatih dengan dataset yang diambil dari Generator, lalu di training untuk membedakan keduanya. Gambar dari Generator yang berhasil di deteksi oleh Diskriminator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas untuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminator, lalu, Diskriminator tidak bisa membedakan fake dan real.

Gambar 8.36 gambaran proses diskriminator network

8.2.2.4 Jelaskan proses training 3D-GANs

Proses training 3D-GANs dilakukan seperti langkah-langkah berikut:

- Terdapat sebuah vektor noise dengan dimensi 200 dari distribusi Gaussian(normal).
- Meng-generate gambar palsu menggunakan model generator.
- Melatih jaringan generator dengan gambar yang asli(sampel dari data yang real) dan dengan gambar palsu yang dihasilkan oleh generator.
- Gunakan adversial model untuk melatih generator model, jangan melatih diskriminator model.
- Ulangi langkah ini dengan jumlah epoch tertentu.

8.2.2.5 Jelaskan bagaimana melakukan settingan awal chapter 02 untuk memenuhi semua kebutuhan sebelum melanjutkan ke tahapan persiapan data.

Persiapkan data 3DShapeNets, setelah itu lakukan seperti dibawah ini:

- membaca buku panduan dan keterangan dari Generative-Adversial-Network project ini. lalu clone github https://github.com/PacktPublishing/Generative-Adversarial-Networks-Projects

- siapkan komputer atau bisa menggunakan google colab
- install library sesuai requirements.txt atau membuat virtual env baru dan install librarynya.

8.2.2.6 Jelaskan tentang dataset yang digunakan, dari mulai tempat unduh, cara membuka dan melihat data. sampai deskripsi dari isi dataset dengan detail penjelasan setiap folder/file yang membuat orang awam paham.

Dataset yang digunakan yaitu 3DShapeNets yang berisi model model bentuk benda, folder train berisi model train dan folder test berisi data model testing. dan semua data tersebut di simpan didalam folder volumetric_data.

8.2.2.7 Jelaskan apa itu voxel dengan ilustrasi dan bahasa paling awam

Secara singkat, voxel dapat diartikan sebagai 3D pixel atau pixel dalam bentuk 3D

Gambar 8.37 Gambar Voxel dan Pixel

```
#In[]:Testing Display
import scipy.io as io
voxels = io.loadmat("/content/3DShapeNets/volumetric_data/door/30/
    test/door_000000011_1.mat")['instance']

import numpy as np
voxels = np.pad(voxels, (1, 1), 'constant', constant_values=(0, 0))

import scipy.ndimage as nd
voxels = nd.zoom(voxels, (2, 2, 2), mode='constant', order=0)

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

```
fig = plt.figure()
ax = Axes3D(fig)
ax.voxels(voxels, edgecolor="red")

plt.show()
plt.savefig('/content/drive/My Drive/zMachine Learning/3DShape/data')
```

Kode di atas befungsi untuk visualisasidataset dalam tampilan plot. langkah-langkah seperti ini : import library, load data file .mat dan lakukan read memakai matplotlib, Hasilnya adalah sebagai berikut:

Gambar 8.38 Gambar Voxel dan Pixel

8.2.2.9 Buka file run.py jelaskan perbaris kode pada fungsi untuk membuat generator yaitu build generator

```
def build_generator():
    """
    Create a Generator Model with hyperparameters values defined as
    follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
```

```
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5

    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model
```

Kode di atas befungsi untuk membuat generator yaitu dengan ketentukan gen sebagai variabel dan membuat fungsi atau variabel genmodel lalu dilakukan return.

8.2.2.10 jelaskan juga fungsi untuk membangun diskriminator pada fungsi build discriminator.

```
def build_discriminator():
    """
    Create a Discriminator Model using hyperparameters values defined
     as follows
    """

    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
               strides=dis_strides[0],
               padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

```

```
    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)

    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model
```

Kode di atas befungsi untuk membangun diskriminator berfungsi untuk mendefenisikan seluruh gambar yang sudah di load generator sebagai gambar fake dan real.

8.2.2.11 jelaskan apa maksud dari kode program name == ' main '

```
if __name__ == '__main__':
```

Jika interpreter python menjalankan if name == main sebagai program utama, itu ialah menetapkan variabel name untuk memiliki nilai main. Jika file ini sedang di impor dari modul lain, name akan ditetapkan ke nama modul. Nama modul tersedia sebagai nilai untuk name variabel global.

8.2.2.12 jelaskan secara detil perbaris dan per parameter apa arti dari kode program :

```
#In[]: Soal 12
    """
    Specify Hyperparameters
    """
    object_name = "chair" #Membuat variable
    data_dir = "/content/3DShapeNets/volumetric_data/" \
               "{}/30/train/*.mat".format(object_name) #Memasukkan
    direktori dari data training
    gen_learning_rate = 0.0025 #Membuat rate pada generator
    dis_learning_rate = 10e-5 #Membuat rate pada diskriminator
    beta = 0.5 #Menentukan beta
    batch_size = 1 #Menentukan ukuran batch
    z_size = 200 #Menentukan ukuran data training
    epochs = 10 #Menentukan masa periode. setiap epoch memiliki 1
    batch
    MODE = "train" #Menentukan variable isinya train
```

Kode di atas befungsi untuk melakukan load dataset dengan ketentuan data yang hanya dalam folder chair pada data train.

8.2.2.13 Jelaskan secara detil dari kode program pembuatan dan kompilasi arsitektur berikut :

```
#In[]: Soal 13
    """
    Create models
    """
    gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
    dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)

    discriminator = build_discriminator()
    discriminator.compile(loss='binary_crossentropy', optimizer=
    dis_optimizer)

    generator = build_generator()
    generator.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

Kode di atas menggunakan Adam sebagai algoritma pengoptimalan dan binary_crossentropy sebagai kerugian loss.

8.2.2.14 Jelaskan secara detil kode program untuk membuat dan melakukan kompilasi model adversarial berikut:

```
    discriminator.trainable = False

    input_layer = Input(shape=(1, 1, 1, z_size))
    generated_volumes = generator(input_layer)
    validity = discriminator(generated_volumes)
    adversarial_model = Model(inputs=[input_layer], outputs=[validity
    ])
    adversarial_model.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

Kode di atas artinya ialah kita memasukkan random vector kedalam generator model lalu membagi 2 yaitu generated example dan real example, dan meneruskan ke diskriminator model sebagai real atau fake

8.2.2.15 Jelaskan Ekstrak dan load data kursi dengan menggunakan fungsi getVoxelsFormat dan get3DImages yang digunakan pada kode program berikut :

```
#In[]: Soal 15
    print("Loading data...")
    volumes = get3DImages(data_dir=data_dir)
    volumes = volumes[..., np.newaxis].astype(np.float)
    print("Data loaded...")
```

Kode di atas befungsi untuk melakukan load data pada dataset.

8.2.2.16 Jelaskan maksud dari kode program instansiasi TensorBoard yang menambahkan generator dan diskriminator pada program berikut:

```
#In[]: Soal 16
```

```
    tensorboard = TensorBoard(log_dir="/content/drive/My Drive/
    zMachine Learning/3DShape/logs/{}".format(time.time()))
    tensorboard.set_model(generator)
    tensorboard.set_model(discriminator)
```

Kode di atas berfungsi untuk membuat tensorboard untuk mencatat log yang akan dihasilkan dari training data kita.

8.2.2.17 Jelaskan apa fungsi dari np reshape ones zeros pada kode program berikut dengan parameternya:

```
#In[]: Soal 17
    labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1, 1, 1)
    )
    labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1, 1,
    1))
```

Kode di atas befungsi untuk melakukan reshape agar shape yang dihasilkan tidak terlalu besar. Dengan membuat variabel real dan fake.

8.2.2.18 Jelaskan kenapa harus ada perulangan dalam meraih epoch. Dan jelaskan apa itu epoch terkait kode program berikut:

```
#In[]: Soal 18
    if MODE == 'train':
        for epoch in range(epochs):
            print("Epoch:", epoch)

            gen_losses = []
            dis_losses = []
```

Kode di atas befungsi untuk melakukan training epoch, karena jika epoch semakin banyak maka kualiatas training yang dihasilkan akan semakin baik dan training epoch akan dilakukan jika variable mode berisikan train.

8.2.2.19 Jelaskan apa itu batches dan kaitannya dengan kode program berikut, dan kenapa berada di dalam epoch:

```
#In[]: Soal 19
            number_of_batches = int(volumes.shape[0] / batch_size)
            print("Number of batches:", number_of_batches)
            for index in range(number_of_batches):
                print("Batch:", index + 1)
```

Batch adalah jumlah file yang akan di training.

8.2.2.20 Berikut adalah kode program pengambilan gambar dan noise. Jelaskan apa fungsi np.random.normal serta astype, serta jelaskan apa arti parameter titik dua dan jelaskan isi dari z sample dan volumes batch:

```
#In[]: Soal 20
                z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]
```

Kode di atas befungsi untuk gambar menjadi bersih dari noise dan juga menyesuaikan shapenya.

8.2.2.21 Berikut adalah kode program generator gambar palsu. Jelaskan apa fungsi generator.predict on batch, serta jelaskan apa arti parameter z sample:

```
#In[]: Soal 21
                # Next, generate volumes using the generate network
                gen_volumes = generator.predict_on_batch(z_sample)
```

Kode di atas befungsi untuk membuat sample gambar atau mempredict gambar yang nantinya akan diteruskan ke diskriminator.

8.2.2.22 Berikut adalah kode program training diskriminator dengan gambar palsu dari generator dan gambar asil. Jelaskan apa maksudnya harus dilakukan training diskriminator secara demikian dan jelaskan apa isi loss fake dan loss real serta d loss dan fungsi train on batch.

```
#In[]: Soal 22
                """
                Train the discriminator network
                """
                discriminator.trainable = True
                if index % 2 == 0:
                    loss_real = discriminator.train_on_batch(volumes_batch, labels_real)
                    loss_fake = discriminator.train_on_batch(gen_volumes, labels_fake)

                    d_loss = 0.5 * np.add(loss_real, loss_fake)
                    print("d_loss:{}".format(d_loss))

                else:
                    d_loss = 0.0

                discriminator.trainable = False
```

Kode di atas befungsi untuk membuat diskriminator bisa load gambar fake dan real dari generator, oleh karena itu ada generator loss dan diskriminator loss untuk melihat seberapa baik kualitas yang dihasilkan.

8.2.2.23 Berikut adalah kode program training model adversarial yang terdapat generator dan diskriminator. Jelaskan apa bagaimana proses terbentuknya parameter z dan g loss:

```
#In[]: Soal 23
                    """
                    Train the generator network
                    """
                    z = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                    g_loss = adversarial_model.train_on_batch(z, labels_real)
                    print("g_loss:{}".format(g_loss))

                    gen_losses.append(g_loss)
                    dis_losses.append(d_loss)
```

Kode di atas befungsi untuk melakukan print gloss untuk generator dan juga dloss untuk diskriminator.

8.2.2.24 Berikut adalah kode program generate dan menyimpan gambar 3D setelah beberapa saat setiap epoch. Jelaskan mengapa ada perulangan dengan parameter tersebut, serta jelaskan arti setiap variabel beserta perlihatkan isinya dan artikan isinya :

```
#In[]: Soal 24
                    # Every 10th mini-batch, generate volumes and save them
                    if index % 10 == 0:
                        z_sample2 = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                        generated_volumes = generator.predict(z_sample2, verbose=3)
                        for i, generated_volume in enumerate(generated_volumes[:5]):
                            voxels = np.squeeze(generated_volume)
                            voxels[voxels < 0.5] = 0.
                            voxels[voxels >= 0.5] = 1.
                            saveFromVoxels(voxels, "/content/drive/My Drive/zMachine Learning/3DShape/results/img_{}_{}_{}".format(epoch, index, i))
```

melakukan perulangan dimana setiap terdapat 10 mini-batch akan meng-generate volumenya dan akan menyimpannya lalu membandingkannya.

8.2.2.25 Berikut adalah kode program menyimpan average losses setiap epoch. Jelaskan apa itu tensorboard dan setiap parameter yang digunakan pada kode program ini :

```
#In[]: Soal 25
            # Write losses to Tensorboard
            write_log(tensorboard, 'g_loss', np.mean(gen_losses), epoch)
            write_log(tensorboard, 'd_loss', np.mean(dis_losses), epoch)
```

menuliskan log losses ke Tensorboard agar disimpan

8.2.2.26 Berikut adalah kode program menyimpan model. Jelaskan apa itu format h5 dan penjelasan dari kode program berikut :

```
#In[]: Soal 26
        """
        Save models
        """
        generator.save_weights(os.path.join("models", "generator_weights.h5"))
        discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))
```

untuk menyimpan berat model generator dan model diskriminator kedalam file h5.

H5 adalah Hierarchical Data Format 5 File. File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah. File H5 biasanya digunakan di luar angkasa, fisika, teknik, keuangan, penelitian akademis, genomik, astronomi, instrumen elektronik, dan bidang medis.

8.2.2.27 Berikut adalah kode program testing model. Jelaskan dengan ilustrasi gambar dari mulai meload hingga membuat gambar 3D dengan menggunakan z sample, bisakah parameter z sample tersebut diubah2? :

```
#In[]: Soal 27
    if MODE == 'predict':
        # Create models
        generator = build_generator()
        discriminator = build_discriminator()

        # Load model weights
        generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
        discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)

        # Generate 3D models
        z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
        generated_volumes = generator.predict(z_sample, verbose=3)

        for i, generated_volume in enumerate(generated_volumes[:2]):
            voxels = np.squeeze(generated_volume)
            voxels[voxels < 0.5] = 0.
            voxels[voxels >= 0.5] = 1.
            saveFromVoxels(voxels, "/content/drive/My Drive/zMachine Learning/3DShape/results/gen_{}".format(i))
```

untuk membuat mode predict/prediksi. dimana dia melakukan pebuatan model generator, model diskriminator dan meload data h5 yang sudah dibuat tadi dan menggenerate nya kedalam model 3D, lalu menyimpan model voxel nya ke folder results.

Gambar 8.39 Salah satu hasil

8.2.3 Penanganan Error

8.2.3.1 Error

- OSError

Gambar 8.40 OSError

8.2.3.2 Solusi Error

- OSError

  Pastikan folder telah ada

8.2.4 Bukti Tidak Plagiat

Gambar 8.41 Bukti tidak plagiat

8.2.5 Link Youtube

https://bit.ly/KecerdasanBuatanDirga

8.3 1174087 - Ilham Muhammad Ariq

8.3.1 Teori

8.3.1.1 Apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.

Generator adalah pembuat sesuatu atau dapat menghasilkan sesuatu. Jika perumpamaannya mahasiswa maka seperti ini :

Gambar 8.42 No 1

8.3.1.2 Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatornya.

Arti kata diskriminator adalah rangkaian dalam berbagai alat. Berasal dari kata diskriminasi yang merujuk pada pelayanan yang tidak adil terhadap individu tertentu, di mana layanan ini dibuat berdasarkan karakteristik yang diwakili oleh individu tersebut. dalam machine learning, Discrimination itu bisa ada dua, yaitu:

1. Disparate Treatment(perawatan berbeda) atau membedakan/ mengklasifikasikan sesuatu atau seseorang

2. Disparate Impact (dampak berbeda) ialah melihat konsekuensi klasifikasi/ pengambilan keputusan pada kelompok tertentu.

Gambar 8.43 No 2

8.3.1.3 Bagaimana arsitektur generator dibuat

Aristektur generator didalam GAN terdiri dari lima lapisan, yaitu: satu input layer, tiga dense layer dan satu output layer.

Gambar 8.44 Arsitektur generator pada GAN-Project

Dengan ilustrasi sebagai berikut:

Gambar 8.45 No 3

8.3.1.4 Bagaimana arsitektur diskriminator dibuat

aristektur diskriminator didalam GAN terdiri dari lima lapisan, yaitu: satu input layer, tiga dense layer dan satu output layer. Diskriminator pada GAN-Project:

Gambar 8.46 Arsitektur Diskriminator pada GAN-Project

Dengan ilustrasi sebagai berikut:

Gambar 8.47 No 4

8.3.1.5 Apa itu latent space.

Latent space adalah space atau spot yang tersembunyi, sehingga akan mengambil titik data yang dekat dan yang mirip. Dengan ilustrasi sebagai berikut:

Gambar 8.48 No 5

8.3.1.6 Apa itu adversarial play

Dalam GAN adversial play ialah persaingan antara generator dan diskriminator.

Gambar 8.49 No 6

8.3.1.7 Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium

Nash equilibrium adalah strategi yang dipilih oleh masing-masing pemain, dengan strategi pemain lain tertentu. Strategi ini merupakan strategi terbaik dari masing-masing pemain dengan mempertimbangkan strategi tertentu yang diambil oleh pemain lainnya, atau yang disebut dengan Best Respons.

SAM \ TOM	A	B
A	1,1	1,-1
B	-1,1	0,0

Gambar 8.50 No 7

8.3.1.8 Sebutkan dan jelaskan contoh-contoh implementasi dari GAN

Contoh implementasi dari GAN salah satunya adalah men-generate wajah kucing

Gambar 8.51 No 8

8.3.1.9 Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer)

Gambar 8.52 Arsitektur generator pada GAN-Project

```
def generator_model():
    model = Sequential([
        Dense(1024, input_dim=100, activation='tanh'),
        Dense(128*7*7),
        Reshape((7, 7, 128)),
        UpSampling2D(size=(2, 2)),
        Conv2D(64, (5, 5), padding='same', activation='tanh'),
        UpSampling2D(size=(2, 2)),
        Conv2D(1, (5, 5), padding='same', activation='tanh')
    ])
    return model

generator_model().summary()
```

Listing 8.22 Kode program arsitektur generator

8.3.1.10 Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuath input layer, 3 buah hidden layer, dan satu output layer.

Gambar 8.53 Arsitektur Diskriminator pada GAN-Project

```
def discriminator_model():
    model = Sequential([
        Conv2D(64, (5, 5), padding='same', input_shape=(28, 28, 1),
    activation='tanh'),
        MaxPooling2D(pool_size=(2, 2)),
        Conv2D(128, (5, 5),activation='tanh'),
        MaxPooling2D(pool_size=(2, 2)),
        Flatten(),
        Dense(1024, activation='tanh'),
        Dense(1, activation='sigmoid')
    ])
    return model

discriminator_model().summary()
```

Listing 8.23 Kode program arsitektur diskriminator

8.3.1.11 Kaitan output dan input antara generator dan diskriminator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

Kaitan antara output pada generator dan inputan pada diskriminator adalah suatu keterhubungan yang akan menghasilkan data yang benar, karena output dari generator merupakan inoutan pada diskriminator

Gambar 8.54 No 9

8.3.1.12 Perbedaan antara Kullback-Leibler divergence (KL divergence)/relative entropy, Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to the average dalam mengukur kualitas dari model.

diantara keduanya yang paling terkenal adalah KL divergence, karena KL divergence memiliki beberaa sifat/rumus yang bagus .salah satunya adalah KL[q; p] jenis daerah yang di mana q(x) memiliki massa non-null dan p(x) memiliki massa null. Ini mungkin terlihat seperti bug, tetapi sebenarnya ini merupakan fitur dalam situasi tertentu. sedangkan JS tidak memilikinya,

8.3.1.13 Fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

fungsi objektif merupakan fungsi yang akan mengambil parameter data dan model sebagai argumen, dan dapat dievaluasi untuk mengembalikan angka sehingga mampu membedakan gambar yang asli dan yang di buat/fake.

8.3.1.14 Scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

Inception Score digunakan untuk mengukur seberapa realistis output dari GAN, dimana ada dua parameter, yaitu: gambarnya punya variasi dan setiap gambar jelas terlihat seperti sesuatu. Frechet Inception Distance adalah ukuran kesamaan antara dua dataset gambar. Itu terbukti berkorelasi baik dengan penilaian manusia terhadap kualitas visual dan paling sering digunakan untuk mengevaluasi kualitas sampel Generative Adversarial Networks. FID dihitung dengan menghitung jarak Frchet antara dua Gaussians dipasang ke representasi fitur dari jaringan Inception.

8.3.1.15 Kelebihan dan kekurangan GAN

- Kelebihan

  1. GAN Menghasilkan data baru yang bisa hampir mirip dengan data asli. Karena hasil pelatihannya, GAN dapat menghasilkan data gambar, teks, audio, dan video yang dapat dibilang hampir mirip dengan yang aslinya. Berkat hal tersebut, GAN dapat digunakan dalam sistem marketing, e-commerce, games,iklan, dan industri lainnya
  2. GAN mempelajari representasi data secara internal sehingga beberapa masalah pada machine learning dapat diatasi dengan mudah

3. Discriminator yang sudah dilatih dapat menjadi sebuah classifier atau pendeteksi jika data sudah sesuai. Karena Discriminator yang akan menjadi tidak efisien berkat seringnya dilatih
4. GAN dapat dilatih menggunakan data yang belum dilabeled

- Kekurangan

1. Data saat diproses oleh metode gan tidak konvergensi
2. Jenis sampel yang dihasilkan oleh generator terbatas karena modenya terbatas
3. Ketidak seimbangnya antara generator dan discriminator dapat menyebabkan overfitting atau terlalu dekat dengan hasil sampel
4. Sangat sensitif dengan data yang sudah diinisiasi sebelumnya

8.3.2 Praktek

8.3.2.1 Jelaskan apa itu 3D convolutions

3D convolutions merupakan operasi konvolusi 3D yang menerapkan filter 3D ke data input dengan tiga arah, yaitu x,y, dan z. fitur ini menciptakan sebuah daftar peta yang ditumpuk. agar lebih mudah saya menggunakan sebuah ilustrasi seperti berikut:

Gambar 8.55 gambaran 3D Convolutions

8.3.2.2 Jelaskan dengan kode program arsitektur dari generator networknya, beserta penjelasan input dan output dari generator network.

```
def build_generator():#nama kelasnya
    """
    Create a Generator Model with hyperparameters values defined as
    follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5
```

```
    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model
```

kode diatas merupakan generator network yang terdiri dari lima layer, satu input layer, tiga hidden layer, dan satu output layer.Dan prosesnya dapat digambarkan seperti berikut:

Gambar 8.56 gambaran proses generator network

penjelasan

- Input layer mengambill 100-dimensi sampel dari distribusi Gaussian(normal) dan meneruskan tensor ke. hidden layer pertama tanpa ada modifikasi
- ketiga hidden layer adalah dense layer dengan unit masing-masing 500, 500, dan 784. dimana dense layer pertama mengkonversi bentuk tensor (batch_size, 100) ke bentuk tensor(batch_size, 500). sedangkan pada layer dense kedua menghasilkan bentuk tensor (batch_size, 500), dan layer dense ketiga menghasilkan (batch_size, 784)
- Output layer, tensor akan dibentuk kembali dari bentuk tensor (batch_size, 784) menjadi (batch_size, 28, 28) artinya hasil dari generator ini akan menghasilkan banyak gambar, dimana setiap gambarnya memiliki ukuran (28, 28).

8.3.2.3 Jelaskan dengan kode program arsitektur dari diskriminator network, beserta penjelasan input dan outputnya.

```
def build_discriminator():
    """
    Create a Discriminator Model using hyperparameters values defined
     as follows
    """
    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
               strides=dis_strides[0],
               padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)

    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model
```

kode diatas merupakan diskriminator network, dimana prosesnya dapat digambarkan seperti berikut:

Gambar 8.57 gambaran proses diskriminator network

penjelasan

- awalnya diskriminator menerima input dengan bentuk 28x28.
- Input layer mengambil tensor input dan meneruskannya ke hidden layer pertama tanpa modifikasi apapun.
- lalu flattens layer akan meratakan tensor menjadi 784-dimensi vektor, yang akan diteruskan ke hidden layer (dense layer) pertama. hidden layer pertama dan kedua aka memodifikasinya menjadi vektor 500-dimensi.
- Output layer mesuk kedalam dense layer, dengan satu unit neuron dan sigmoid sebagai fungsi aktivasi. sigmoid akan menghasilkan nilai tunggal, 0 atau 1. Nilai 0 akan menunjukan bahwa gambar yang diberikan palsu. sebaliknya jika nilai 1 maka gambar asli.

8.3.2.4 Jelaskan proses training 3D-GANs

Proses training 3D-GANs dilakukan seperti langkah-langkah berikut:

- Terdapat sebuah vektor noise dengan dimensi 200 dari distribusi Gaussian(normal).
- Meng-generate gambar palsu menggunakan model generator.
- Melatih jaringan generator dengan gambar yang asli(sampel dari data yang real) dan dengan gambar palsu yang dihasilkan oleh generator.
- Gunakan adversial model untuk melatih generator model, jangan melatih diskriminator model.
- Ulangi langkah ini dengan jumlah epoch tertentu.

8.3.2.5 Jelaskan bagaimana melakukan settingan awal chapter 02 untuk memenuhi semua kebutuhan sebelum melanjutkan ke tahapan persiapan data.

Sebelum melakukan tahapan persiapan data, kita harus melakukan beberapa hal seperti berikut:

- membaca buku panduan atau keterangan dari Generative-Adversial-Network project ini. bisa didowload di portal kampus keren atau di github https://github.com/PacktP Adversarial-Networks-Projects
- persiapkan laptop/pc dengan spek yang cukup bagus(tidak kentang). jika tidak ada, bisa menggunakan google colab(cara penggunaanya akan dijelaskan di video).
- usahakn versi yang terinstall sama dengan versi yang ada pada requirement.txt agar terhindar dari error.

8.3.2.6 Jelaskan tentang dataset yang digunakan, dari mulai tempat unduh, cara membuka dan melihat data. sampai deskripsi dari isi dataset dengan detail penjelasan setiap folder/file yang membuat orang awam paham.

Penjelasan dataset

- dataset bisa didownload pada halaman: http://3dshapenets.cs.princeton.edu/3DShapeNets
- Untuk membuka data, setelah didownload cukup di eksrak saja menggunakan winrar/ 7zip
- Untuk mengetahui lebih jelasnya tentang dataset ini bisa dibuka saja file README.md, disana akan dijelaskan semuanya.

8.3.2.7 Jelaskan apa itu voxel dengan ilustrasi dan bahasa paling awam

Secara singkat, voxel dapat diartikan sebagai 3D pixel atau pixel dalam bentuk 3D. seperti pada ilustrasi berikut:

Gambar 8.58 Pixel VS Voxel

8.3.2.8 Visualisasikan dataset tersebut dalam tampilan visual plot, jelaskan cara melakukan visualisasinya

```
import scipy.io as io
import matplotlib.pyplot as plt
import scipy.ndimage as nd
import numpy as np
from mpl_toolkits.mplot3d import Axes3D

voxels = io.loadmat("3DShapeNets/volumetric_data/guitar/30/test/
    guitar_000000003_1.mat")['instance']
voxels = np.pad(voxels, (1, 1), 'constant', constant_values=(0, 0))
voxels = nd.zoom(voxels, (2, 2, 2), mode='constant', order=0)

fig = plt.figure()
ax = Axes3D(fig)
ax.voxels(voxels, edgecolor="chartreuse")

plt.show()
plt.savefig('guitar')
```

Kode diatas berfungsi untuk melakukan visualisasi dataset dalam tampilan plot. dengan tahapan import library, load data file .mat, dan lakukan read memakai matplotlib. dan hasilnya seperti berikut

Gambar 8.59 Visualisasi dataset(contoh)

8.3.2.9 Buka file run.py jelaskan perbaris kode pada fungsi untuk membuat generator yaitu build generator

```
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5
```

potongan kode diatas berfungsi sebagai penentuan nilai untuk hyperparameters yang berbeda.

```
input_layer = Input(shape=gen_input_shape)
```

potongan kode diatas berfungsi untuk membuat input layer

```
# First 3D transpose convolution(or 3D deconvolution) block
a = Deconv3D(filters=gen_filters[0],
             kernel_size=gen_kernel_sizes[0],
             strides=gen_strides[0])(input_layer)
a = BatchNormalization()(a, training=True)
a = Activation(activation='relu')(a)
```

potongan kode diatas berfungsi untuk menambahkan 3D transpose pertama.

```
# Next 4 3D transpose convolution(or 3D deconvolution) blocks
for i in range(gen_convolutional_blocks - 1):
    a = Deconv3D(filters=gen_filters[i + 1],
                 kernel_size=gen_kernel_sizes[i + 1],
                 strides=gen_strides[i + 1], padding='same')(a)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation=gen_activations[i + 1])(a)
```

potongan kode diatas berfungsi untuk menambahkan empat 3D transpose lainya.

```
gen_model = Model(inputs=[input_layer], outputs=[a])
```

potongan kode diatas berfungsi untuk membuat model keras dan menentukan input dan output untuk network generator.

8.3.2.10 jelaskan juga fungsi untuk membangun diskriminator pada fungsi build discriminator.

```
dis_input_shape = (64, 64, 64, 1)
dis_filters = [64, 128, 256, 512, 1]
dis_kernel_sizes = [4, 4, 4, 4, 4]
dis_strides = [2, 2, 2, 2, 1]
dis_paddings = ['same', 'same', 'same', 'same', 'valid']
dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
```

potongan kode diatas berfungsi sebagai penentuan nilai untuk hyperparameters yang berbeda.

```
dis_input_layer = Input(shape=dis_input_shape)
```

potongan kode diatas berfungsi untuk membuat input layer, berupa gambar 3D dengan dimensi 64x64x64x1

```
# The first 3D Convolutional block
a = Conv3D(filters=dis_filters[0],
           kernel_size=dis_kernel_sizes[0],
           strides=dis_strides[0],
           padding=dis_paddings[0])(dis_input_layer)
# a = BatchNormalization()(a, training=True)
a = LeakyReLU(dis_alphas[0])(a)
```

potongan kode diatas berfungsi untuk menambahkan 3D transpose pertama.

```
    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)
```

potongan kode diatas berfungsi untuk menambahkan empat 3D transpose lainya.

```
    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
```

potongan kode diatas berfungsi untuk membuat model keras dan menentukan input dan output untuk network generator.

8.3.2.11 jelaskan apa maksud dari kode program name == ' main '

```
if __name__ == '__main__':
```

Listing 8.24 Kode program run.py

maksudnya kode tersebut memiliki fungsi jika interpreter python menjalankan kode tersebut, maka akan menetapkan variable name untuk memiliki nilai main. jika file ini di import dari modul lain, maka name akan ditetapkan ke nama modul tersbut.

8.3.2.12 jelaskan secara detil perbaris dan per parameter apa arti dari kode program :

Penjelasan kode:

```
# In[soal 12]
    object_name = "airplane" #membuat variabel
    data_dir = "3DShapeNets/volumetric_data/" \
               "{}/30/train/*.mat".format(object_name) #menunjukan
    direktori dari variable sebelumnya
    gen_learning_rate = 0.0025 #membuat rate pada generator
    dis_learning_rate = 10e-5 #membuat rate pada diskriminator
    beta = 0.5 #
    batch_size = 1 #menentukan ukuran kumpulan gambar
    z_size = 200 #
    epochs = 1 #menetukkan masa periode. setiap epoch memiliki semua
    batch
    MODE = "train" #membuat traon mode
```

Listing 8.25 Kode program run.py

8.3.2.13 Jelaskan secara detil dari kode program pembuatan dan kompilasi arsitektur berikut :

```
    gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
    dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)

    discriminator = build_discriminator()
    discriminator.compile(loss='binary_crossentropy', optimizer=
    dis_optimizer)

    generator = build_generator()
    generator.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

Listing 8.26 Kode program run.py

penjelasanya ialah membuat model dari generator model dan diskriminator model,

8.3.2.14 Jelaskan secara detil kode program untuk membuat dan melakukan kompilasi model adversarial berikut:

```
# In[soal 14]
    discriminator.trainable = False

    input_layer = Input(shape=(1, 1, 1, z_size))
    generated_volumes = generator(input_layer)
    validity = discriminator(generated_volumes)
    adversarial_model = Model(inputs=[input_layer], outputs=[validity
    ])
    adversarial_model.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

Listing 8.27 Kode program run.py

penjelasanya ialah membuat model diskriminator tidak di training dan membuat model adversarial.

8.3.2.15 Jelaskan Ekstrak dan load data kursi dengan menggunakan fungsi getVoxelsFormat dan get3DImages yang digunakan pada kode program berikut :

```
# In[soal 15]
    print("Loading data...")
    volumes = get3DImages(data_dir=data_dir)
    volumes = volumes[..., np.newaxis].astype(np.float)
    print("Data loaded...")
```

Listing 8.28 Kode program run.py

penjelasanya ialah melakukan print loading data, membuat variabel volumes untuk mendapatkan data gamabr 3D.

8.3.2.16 Jelaskan maksud dari kode program instansiasi TensorBoard yang menambahkan generator dan diskriminator pada program berikut:

```
# In[soal 16]
    tensorboard = TensorBoard(log_dir="logs/{}".format(time.time()))
    tensorboard.set_model(generator)
    tensorboard.set_model(discriminator)
```

Listing 8.29 Kode program run.py

penjelasanya ialah membuat variabel tensorboard untuk melakukan pencatatan log, dan men-set model generator dan model diskriminator

8.3.2.17 Jelaskan apa fungsi dari np reshape ones zeros pada kode program berikut dengan parameternya:

Penjelasan kode:

```
# In[soal 17]
    labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1, 1, 1)
    )
    labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1, 1,
    1))
```

Listing 8.30 Kode program run.py

membuat label_real untuk menetapkan bahwa jika output 1 itu menunjukan gambar real, dan label_fake untuk menunjukan bahwa output 0 untuk data fake/palsu.

8.3.2.18 Jelaskan kenapa harus ada perulangan dalam meraih epoch. Dan jelaskan apa itu epoch terkait kode program berikut:

```
# In[soal 18]
    if MODE == 'train':
        for epoch in range(epochs):
            print("Epoch:", epoch)

            gen_losses = []
            dis_losses = []
```

Listing 8.31 Kode program run.py

melakukan perulangan jika MODE = train maka akan diulangi terus menerus sampai epoch(periode) terpenuhi dengan disertai keterangan generator dan diskriminator losses.

8.3.2.19 Jelaskan apa itu batches dan kaitannya dengan kode program berikut, dan kenapa berada di dalam epoch:

```
# In[soal 19]
            number_of_batches = int(volumes.shape[0] / batch_size)
            print("Number of batches:", number_of_batches)
            for index in range(number_of_batches):
                print("Batch:", index + 1)
```

Listing 8.32 Kode program run.py

menghitung jumlah batch yang telah ditrain

8.3.2.20 Berikut adalah kode program pengambilan gambar dan noise. Jelaskan apa fungsi np.random.normal serta astype, serta jelaskan apa arti parameter titik dua dan jelaskan isi dari z sample dan volumes batch:

```
# In[soal 20]
                z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]
```

Listing 8.33 Kode program run.py

untuk membersihkan gambar dari noise dan menyesuaikan shape.

8.3.2.21 Berikut adalah kode program generator gambar palsu. Jelaskan apa fungsi generator.predict on batch, serta jelaskan apa arti parameter z sample:

```
# In[soal 21]
                # Next, generate volumes using the generate network
                gen_volumes = generator.predict_on_batch(z_sample)
```

Listing 8.34 Kode program run.py

untuk meng-generate volume menggunakan model jaringan generator.

8.3.2.22 Berikut adalah kode program training diskriminator dengan gambar palsu dari generator dan gambar asil. Jelaskan apa maksudnya harus dilakukan training diskriminator secara demikian dan jelaskan apa isi loss fake dan loss real serta d loss dan fungsi train on batch.

```
# In[soal 22]
                discriminator.trainable = True
                if index % 2 == 0:
                    loss_real = discriminator.train_on_batch(volumes_batch, labels_real)
                    loss_fake = discriminator.train_on_batch(gen_volumes, labels_fake)

                    d_loss = 0.5 * np.add(loss_real, loss_fake)
                    print("d_loss:{}".format(d_loss))

                else:
                    d_loss = 0.0
```

Listing 8.35 Kode program run.py

untuk membuat diskriminator bisa meload gambar fake dan real dari model generator. dengan disertai genrator dan diskriminator loss, agar terliaht seberapa baik kualitas yang dihasilkan.

8.3.2.23 Berikut adalah kode program training model adversarial yang terdapat generator dan diskriminator. Jelaskan apa bagaimana proses terbentuknya

parameter z dan g loss:

```
# In[soal 23]
                discriminator.trainable = False
                """
                Train the generator network
                """
                z = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                g_loss = adversarial_model.train_on_batch(z, labels_real)
                print("g_loss:{}".format(g_loss))

                gen_losses.append(g_loss)
                dis_losses.append(d_loss)
```

Listing 8.36 Kode program run.py

untuk mentrain model generator dan variabel g_loss untuk melakukan perbandingan antara label gambar yang asli.

8.3.2.24 Berikut adalah kode program generate dan menyimpan gambar 3D setelah beberapa saat setiap epoch. Jelaskan mengapa ada perulangan dengan parameter tersebut, serta jelaskan arti setiap variabel beserta perlihatkan isinya dan artikan isinya :

```
# In[soal 24]
                # Every 10th mini-batch, generate volumes and save them
                if index % 10 == 0:
                    z_sample2 = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                    generated_volumes = generator.predict(z_sample2, verbose=3)
                    for i, generated_volume in enumerate(generated_volumes[:5]):
                        voxels = np.squeeze(generated_volume)
                        voxels[voxels < 0.5] = 0.
                        voxels[voxels >= 0.5] = 1.
                        saveFromVoxels(voxels, "results/img_{}_{}_{}".format(epoch, index, i))
```

Listing 8.37 Kode program run.py

untuk melakukan perulangan dimana setiap terdapat 10 mini-batch akan meng-generate volumenya dan akan menyimpannya.

8.3.2.25 Berikut adalah kode program menyimpan average losses setiap epoch. Jelaskan apa itu tensorboard dan setiap parameter yang digunakan pada kode program ini :

```
# In[soal 25]
```

```
            # Write losses to Tensorboard
            write_log(tensorboard, 'g_loss', np.mean(gen_losses), epoch)
            write_log(tensorboard, 'd_loss', np.mean(dis_losses), epoch)
```

Listing 8.38 Kode program run.py

untuk menuliskan log losses ke Tensorboard

8.3.2.26 Berikut adalah kode program menyimpan model. Jelaskan apa itu format h5 dan penjelasan dari kode program berikut :

```
# In[soal 26]
        generator.save_weights(os.path.join("models", "generator_weights.h5"))
        discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))
```

Listing 8.39 Kode program run.py

untuk menyimpan berat model generator dan model diskriminator kedalam file h5.

H5 adalah Hierarchical Data Format 5 File. File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah. File H5 biasanya digunakan di luar angkasa, fisika, teknik, keuangan, penelitian akademis, genomik, astronomi, instrumen elektronik, dan bidang medis.

8.3.2.27 Berikut adalah kode program testing model. Jelaskan dengan ilustrasi gambar dari mulai meload hingga membuat gambar 3D dengan menggunakan z sample, bisakah parameter z sample tersebut diubah2? :

```
        generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
        discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)

        # Generate 3D models
        z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
        generated_volumes = generator.predict(z_sample, verbose=3)

        for i, generated_volume in enumerate(generated_volumes[:2]):
            voxels = np.squeeze(generated_volume)
            voxels[voxels < 0.5] = 0.
            voxels[voxels >= 0.5] = 1.
            saveFromVoxels(voxels, "results/gen_{}".format(i))
```

Listing 8.40 Kode program run.py

untuk membuat mode predict/prediksi. dimana dia melakukan pebuatan model generator, model diskriminator dan meload data h5 yang sudah dibuat tadi dan menggenerate nya kedalam model 3D, lalu menyimpan model voxel nya ke folde results.

8.3.3 Penanganan Error

1. SS Error

```
FileNotFoundError: [Errno 2] No such file or directory: '3DShapeNets/volumetric_data/guitar/30/test/
guitar_000000003_1.mat'
```

Gambar 8.60 File Not Found error

2. Jenis Error

   - File Not Found Error

3. Cara Penanganan
   Dengan cara menyesuaikan letak file yang ingin dibaca / digunakan

8.3.4 Bukti Tidak Plagiat

Gambar 8.61 Bukti tidak plagiat

8.3.5 Link Youtube

https://youtu.be/hW96rJM9VD8

8.4 1174069 - Fanny Shafira Damayanti

8.4.1 Soal Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.
   Tugas Generator sekarang sedang dibuat untuk membuat koleksi gambar palsu, yang saat ini dilihat oleh Diskriminator. Diskriminator tidak dapat membedakan antara yang asli dan yang palsu. Untuk ilustrasi, lihat gambar berikut:

Gambar 8.62 Teori 1

2. Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatornya.

   Diskriminator adalah CNN yang menerima input gambar yang dimiliki dan menghasilkan angka biner yang meminta input gambar, lalu menghasilkan gambar dari dataset asli atau menghasilkan gambar palsu. Untuk ilustrasi, lihat gambar berikut:

Gambar 8.63 Teori 2

3. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur generator dibuat.

   Aksitektur generator dibuat bisa dijelaskan pada gambar berikut :

Gambar 8.64 Teori 3

4. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminator dibuat.

Aksitektur diskriminator dibuat bisa dijelaskan pada gambar berikut :

Gambar 8.65 Teori 4

5. Jelaskan dengan ilustrasi gambar apa itu latent space.
   Latent space dijelaskan pada gambar berikut :

Gambar 8.66 Teori 5

6. Jelaskan dengan ilustrasi gambar apa itu adversarial play.
   Adversarial play dijelaskan pada gambar berikut :

Gambar 8.67 Teori 6

7. Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium.
   Nash equilibrium adalah Teori permainan adalah studi tentang interaksi strategis antara agen rasional. Sederhananya itu berarti itu adalah studi interaksi ketika pihak-pihak yang terlibat mencoba dan melakukan yang terbaik dari sudut pandang mereka, detailnya dapat dijelaskan pada gambar berikut :

		PRISONER 2	
		Confess	Lie
PRISONER 1	Confess	-8 , -8	0 , -10
	Lie	-10 , 0	-1 , -1

Gambar 8.68 Teori 7

8. Sebutkan dan jelaskan contoh-contoh implementasi dari GAN.

Menurut saya implementasi 3DGAN yaitu pada MAPS dan juga IKEA. Karena pada maps dan juga ikea sudah menerapkan bentuk 3 dimensi yang bisa lebih menarik perhatikan pengguna.

9. Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer).

   Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :

Gambar 8.69 Teori 9

10. Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuath input layer, 3 buah hidden layer, dan satu output layer.

    Untuk penjelasan tersebut dijelaskan pada gambar dibawah ini :

Gambar 8.70 Teori 10

11. Jelaskan bagaimana kaitan output dan input antara generator dan diskriminator tersebut. Jelaskan kenapa inputan dan outputan seperti itu.

    Gambar dari Generator yang berhasil di deteksi oleh Diskriminator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas untuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminator, lalu, Diskriminator tidak bisa membedakan fake dan real.

12. Jelaskan apa perbedaan antara Kullback-Leibler divergence (KL divergence)/relativeentrop
    Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to

the average dalam mengukur kualitas dari model.

Perbedaan nya yaitu memiliki model dari rumus yang berbeda-beda sehingga mempengaruhi hasil train dan test

13. Jelaskan apa itu fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

    Ukuran penting untuk menilai kualitas model. lalu kemudian akan melihat di keseimbangan Nash.

14. Jelaskan apa itu scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

    The inception score adalah algoritma penilaian yang paling banyak digunakan untuk GAN. The Frechet Inception distance adalah Untuk mengatasi berbagai kekurangan Skor awal

15. Jelaskan kelebihan dan kekurangan GAN.

    Kelebihan : GAN dapat memvisualiasikan bentuk model menjadi plot. Kemudian pada Kelemahan : model susah untuk implementasikan yang membuat data training menjadi lemah.

8.4.2 Praktek Program

1. Soal 1

   Konvolusi 3D. Konvolusi 3D menerapkan filter 3 dimensi ke kumpulan data dan filter 3 arah (x, y, z) untuk menghitung representasi fitur tingkat rendah. Bentuk outputnya adalah ruang volume 3 seperti kubus atau berbentuk kubus. 3D sangat membantu dalam pendeteksian peristiwa dalam video, gambar medis 3D, dll. Generative Adversarial Network adalah arsitektur jaringan saraf tiruan yang dimaksudkan untuk membuat atau membuat data yang benar-benar baru, dari nol hingga tidak ada sama sekali. Melihat target utama GAN adalah data gambar. Singkatnya, jaringan GAN berfungsi untuk memberikan gambar baru berdasarkan koleksi gambar yang telah ada sebelumnya selama proses training.

2. Soal 2

```
    Create a Generator Model with hyperparameters values defined
    as follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
```

```
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5

    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(
    a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model

#%% soal10
```

Kode di atas akan melakukan create generator ialah gloss, Bentuk jaringan Generator dapat dilihat berkebalikan dengan struktur jaringan saraf pada umumnya. Generator biasanya menerima input sebuah vektor z, yang kemudian mengubahnya menjadi sebuah output 3D atau 3 dimensi.

3. Soal 3

```
    Create a Discriminator Model using hyperparameters values
    defined as follows
    """

    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
```

```
                strides=dis_strides[0],
                padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)

    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model

#%%
```

Diskrimanator adalah d_loss, Jaringan Discriminator merupakan jaringan klasifikasi biner yang menerima input gambar tiga dimensi dan mengeluarkan klasifikasi menyatakan input gambar adalah gambar asli dari dataset atau merupakan gambar buatan Generator. Diskriminator dilatih dengan dataset yang diambil dari Generator, lalu di training untuk membedakan keduanya. Gambar dari Generator yang berhasil di deteksi oleh Diskriminator sebagai gambar fake, akan dikembalikan dengan feedback pke generator. Kini Generator bertugas untuk bisa membuat sekumpulan gambar palsu, yang nantinya dapat dilihat oleh Diskriminator, lalu, Diskriminator tidak bisa membedakan fake dan real.

4. Soal 4
   Proses training 3D GAN yaitu dengan melakukan epoch sebanyak yang ditentukan.

5. Soal 5

   - Clone github
   - Download dataset
   - Buat folder baru logs dan results

6. Soal 6
   Dataset yang digunakan yaitu 3DShapeNets yang berisi model model bentuk benda dll, folder train berisi train dan folder test berisi data testing. dan semua data tersebut di simpan didalam folder volumetric_data.

7. Soal 7
   Volume pixel atau voxel adalah titik dalam ruang tiga dimensi. Sebuah voxel

mendefinisikan posisi dengan tiga koordinat dalam arah x, y, dan z. Sebuah voxel adalah unit dasar untuk mewakili gambar 3D. Untuk gambar nya ialah sebagai berikut :

Gambar 8.71 Praktek Soal 7

8. Soal 8

```
# -*- coding: utf-8 -*-
"""
Created on Sun May 17 14:56:53 2020

@author: FannyShafira
"""

# In[]

import scipy.io as io
voxels = io.loadmat("data/3DShapeNets/volumetric_data/chair/30/test/chair_000000000_1.mat")['instance']
#%%
import numpy as np
voxels = np.pad(voxels, (1, 1), 'constant', constant_values=(0, 0))
#%%
import scipy.ndimage as nd
voxels = nd.zoom(voxels, (2, 2, 2), mode='constant', order=0)
#%%
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = Axes3D(fig)
ax.voxels(voxels, edgecolor="red")

plt.show()
plt.savefig('data')
```

Kode di atas befungsi untuk visualisasidataset dalam tampilan plot. langkah-langkah seperti ini : import library, load data file .mat dan lakukan read memakai matplotlib, Hasilnya adalah sebagai berikut :

Gambar 8.72 Hasil Soal 8

9. Soal 9

```
def build_generator():
    """
    Create a Generator Model with hyperparameters values defined
    as follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5

    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(
    a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model

#%% soal10
```

Kode di atas befungsi untuk membuat generator yaitu dengan ketentukan gen sebagai variabel dan membuat fungsi atau variabel genmodel lalu dilakukan return.

10. Soal 10

```
def build_discriminator():
    """
    Create a Discriminator Model using hyperparameters values
    defined as follows
    """

    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
               strides=dis_strides[0],
               padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)

    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model

#%%
```

Kode di atas befungsi untuk membangun diskriminator berfungsi untuk mendefenisikan seluruh gambar yang sudah di load generator sebagai gambar fake dan real.

11. Soal 11

Jika interpreter python menjalankan if name == main sebagai program utama, itu ialah menetapkan variabel name untuk memiliki nilai main. Jika file ini

sedang di impor dari modul lain, name akan ditetapkan ke nama modul. Nama modul tersedia sebagai nilai untuk name variabel global.

12. Soal 12

```
    data_dir = "data/3DShapeNets/volumetric_data/" \
               "{}/30/train/*.mat".format(object_name)
    gen_learning_rate = 0.0025
    dis_learning_rate = 10e-5
    beta = 0.5
    batch_size = 1
    z_size = 200
    epochs = 1
    MODE = "train"

#%% soal 13
```

Kode di atas befungsi untuk melakukan load dataset dengan ketentuan data yang hanya dalam folder chair pada data train.

13. Soal 13

```
    Create models
    """
    gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
    dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)

    discriminator = build_discriminator()
    discriminator.compile(loss='binary_crossentropy', optimizer=
    dis_optimizer)

    generator = build_generator()
    generator.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)

#%% soal14
```

Kode di atas menggunakan Adam sebagai algoritma pengoptimalan dan binary_crossentropy sebagai kerugian loss.

14. Soal 14

```

    input_layer = Input(shape=(1, 1, 1, z_size))
    generated_volumes = generator(input_layer)
    validity = discriminator(generated_volumes)
    adversarial_model = Model(inputs=[input_layer], outputs=[
    validity])
    adversarial_model.compile(loss='binary_crossentropy',
    optimizer=gen_optimizer)

#%% soal15
```

Kode di atas artinya ialah kita memasukkan random vector kedalam generator model lalu membagi 2 yaitu generated example dan real example, dan meneruskan ke diskriminator model sebagai real atau fake

15. Soal 15

```
    volumes = get3DImages(data_dir=data_dir)
    volumes = volumes[..., np.newaxis].astype(np.float)
    print("Data loaded...")

#%% soal16
```

Kode di atas befungsi untuk melakukan load data pada dataset.

16. Soal 16

```
    tensorboard.set_model(generator)
    tensorboard.set_model(discriminator)

#%% soal17
```

Kode di atas berfungsi untuk membuat tensorboard yang nantinya bisa diakses melalui localhost.

17. Soal 17

```
    labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1,
    1, 1))

#%% soal18
```

Kode di atas befungsi untuk melakukan reshape agar shape yang dihasilkan tidak terlalu besar. Dengan membuat variabel real dan fake.

18. Soal 18

```
        for epoch in range(epochs):
            print("Epoch:", epoch)

            gen_losses = []
            dis_losses = []

#%% soal19
```

Kode di atas befungsi untuk melakukan training epoch, karena jika epoch semakin banyak maka kualiatas training yang dihasilkan akan semakin baik.

19. Soal 19

```
            print("Number of batches:", number_of_batches)
            for index in range(number_of_batches):
                print("Batch:", index + 1)

#%% soal20
```

Batch adalah jumlah file yang akan di training.

20. Soal 20

```
                volumes_batch = volumes[index * batch_size:(index
    + 1) * batch_size, :, :, :]

#%% soal21
```

Kode di atas befungsi untuk untuk membuat gambar bersih dari noise dan juga menyesuaikan shape.

21. Soal 21

```
                gen_volumes = generator.predict_on_batch(z_sample
    )

                """
```

Kode di atas befungsi untuk membuat sample gambar palsu yang akan diteruskan ke diskriminator.

22. Soal 22

```
                if index % 2 == 0:
                    loss_real = discriminator.train_on_batch(
    volumes_batch, labels_real)
                    loss_fake = discriminator.train_on_batch(
    gen_volumes, labels_fake)

                    d_loss = 0.5 * np.add(loss_real, loss_fake)
                    print("d_loss:{}".format(d_loss))

                else:
                    d_loss = 0.0

#%% soal23
```

Kode di atas befungsi untuk membuat diskriminator bisa load gambar fake dan real dari generator, oleh karena itu ada generator loss dan diskriminator loss untuk melihat seberapa baik kualitas yang dihasilkan.

23. Soal 23

```
                """
                Train the generator network
                """
                z = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                g_loss = adversarial_model.train_on_batch(z, labels_real)
                print("g_loss:{}".format(g_loss))

                gen_losses.append(g_loss)
                dis_losses.append(d_loss)

#%% soal24
```

Kode di atas befungsi untuk melakukan print gloss untuk generator dan juga dloss untuk diskriminator.

24. Soal 24

```
                if index % 10 == 0:
                    z_sample2 = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                    generated_volumes = generator.predict(z_sample2, verbose=3)
                    for i, generated_volume in enumerate(generated_volumes[:5]):
                        voxels = np.squeeze(generated_volume)
                        voxels[voxels < 0.5] = 0.
                        voxels[voxels >= 0.5] = 1.
                        saveFromVoxels(voxels, "results/img_{}_{}_{}".format(epoch, index, i))

#%% soal25
```

Mengapa ada perulangan ? karena untuk melakukan perbandingan dari hasil yang sudah didapat.

25. Soal 25

```
            write_log(tensorboard, 'g_loss', np.mean(gen_losses), epoch)
            write_log(tensorboard, 'd_loss', np.mean(dis_losses), epoch)

        """
```

TensorBoard adalah sebuah aplikasi web localhost untuk memeriksa dan menyelesaikan grafik dari hasil TensorFlow.

26. Soal 26

```
        discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))

#%% soal27
```

File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah.

27. Soal 27

```
        # Create models
        generator = build_generator()
        discriminator = build_discriminator()

        # Load model weights
        generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
        discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)

        # Generate 3D models
        z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
        generated_volumes = generator.predict(z_sample, verbose=3)

        for i, generated_volume in enumerate(generated_volumes[:2]):
            voxels = np.squeeze(generated_volume)
            voxels[voxels < 0.5] = 0.
            voxels[voxels >= 0.5] = 1.
            saveFromVoxels(voxels, "results/gen_{}".format(i))
```

Ini adalah tahap akhir untuk melakukan testing dari model yang telah dibuat dan buat model dari yang sudah di create sebelumnya yaitu generator dan diskriminator. Untuk ilustrasi gambar sebagai berikut :

Gambar 8.73 Praktek Soal 27

8.4.3 Penanganan Error

1. ValueError

Gambar 8.74 ValueError

2. Cara Penanganan Error

   - ValueError
     Error tersebut karena disebabkan gagal load dataset karena salah penamaan direktori.

8.4.4 Bukti Tidak Plagiat

Gambar 8.75 Bukti Tidak Melakukan Plagiat Chapter 8

8.5 1174086 â^ Tia Nur Candida

8.5.1 Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa itu generator dengan perumpamaan anda sebagai mahasiswa sebagai generatornya.

   Generator adalah sebuah jaringan yang merubah inputan vektor menjadi gambar, seperti ada inputan vektor secara acak yaitu angka sembarang, maka angka tersebut akan diubah menjadi gambar yang sembarang pula. Sebagai ilustrasi apabila mahasiswa sebagai generator, mmisalkan inputan vektor tersebut adalah bahan bahan membuat kue, maka hasil dari generator tersebut juga akan acak, bisa kue bolu, cupcake, ataupun kue lainnya.

Gambar 8.76 Teori No 1

2. Jelaskan dengan ilustrasi gambar sendiri apa itu diskriminator dengan perumpamaan dosen anda sebagai diskriminatornya

   Diskriminator adalah sebuah jaringan yang mengeluarkan klasifikasi untuk menyatakan input gambar adalah asli dari dataset atau buatan dari generator. Lebih mudahnya diskriminator digunakan agar hasil gambar dari generator sesuai dengan yang diinginkan. sebagai ilustrasi nya dosen menyuruh mahasiswanya membuat kue bolu, maka hasil yang akan dibuat adalah kue bolu.

Gambar 8.77 Teori No 2

3. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur generator dibua

   Generator menggunkan input array random yang bernama seed, dimana akan diubah menjadi sebuah gambar dengan menggunakan Convolutional Neural Network yang dapat dilihat pada gambar.

Gambar 8.78 Teori No 3

4. Jelaskan dengan ilustrasi gambar sendiri bagaimana arsitektur diskriminator dibuat

   Diskriminator adalah CNN yang menerima inputan image lalu menghasilkan angka biner yang dapat menyatakan apakah gambar tersebut asli atau sesuai dengan dataset asli

Gambar 8.79 Teori No 4

5. Jelaskan dengan ilustrasi gambar apa itu latent space

   latent space adalah representasi dari data yang di kompress, untuk contohnya misalnya ada 3 gambar yaitu 2 kursi yang berebeda dan 1 meja, maka hal hal yang mirip dai kursi tersebut(ciri-ciri utamanya), seeprti itulah latent space

Gambar 8.80 Teori No 5

6. apa itu adversarial play

   adversarial play adalah dimana para jaringan di latih, dimana jaringan satu dan lainnya saling berkompetisi. dapat disimpulkan dimana jaringan generator dan jaringan discriminator saling bertemu berulang ulang kali

7. Jelaskan dengan ilustrasi gambar apa itu Nash equilibrium

   Nash equilibrium adalah konsep dalam teori permainan di mana hasil optimal dari permainan adalah di mana tidak ada insentif untuk menyimpang dari strategi awal mereka. Lebih khusus lagi, keseimbangan Nash adalah konsep teori permainan di mana hasil optimal dari permainan adalah di mana tidak ada pemain yang memiliki insentif untuk menyimpang dari strategi yang dipilihnya setelah mempertimbangkan pilihan lawan.

	A	B
A	1,1	1,-1
B	-1,1	0,0

Gambar 8.81 Teori No 7

8. Sebutkan dan jelaskan contoh-contoh implementasi dari GAN

   Pada bidang mode, seni, dan iklan, GAN dapar digunakan untuk membuat foto-foto model fashion imajinier tanpa perlu menyewa model. Pada bidang sains GAN dapat meningkatkan citra astronomi dan mensimulasikan pelensaan gravitasi untuk penelitian materi gelap.

9. Berikan contoh dengan penjelasan kode program beserta gambar arsitektur untuk membuat generator(neural network) dengan sebuah input layer, tiga hidden layer(dense layer), dan satu output layer(reshape layer)

```
gen=Sequential() #Inisiasi dari sequensial
gen.add(Dense(units=200,input\_dim=np.shape(train\_input)[1]
gen.add(Dense(units=400))#Menambah dense layer dengan batch
gen.add(Dense(units=784, activation='tanh')) #Menambah dense
gen.compile(loss='binary\_crossentropy', optimizer=adam\_opt
 gen.summary() #Memproses data yang sudah disetting dan mena
```

   Pada contoh tersebut, data akan diambil dari hasil proses sebelumnya yaitu proses ekstrasi data gambar. Dari contoh ini, terdapat 3 layer dense, 1 input layer dan 1 output layer. Dari input layer nanti akan dimasukkan terlebih dahulu ke dense layer pertama lalu diproses oleh 2 layer selanjutnya dan terakhir akan ditampilkan oleh layer output.

Gambar 8.82 Teori No 9

10. Berikan contoh dengan ilustrasi dari arsitektur dikriminator dengan sebuath input layer, 3 buah hidden layer, dan satu output layer.

```
diskrim=Sequential()#Inisiasi dari sequensial
diskrim.add(Dense(units=784,input\_dim=np.shape(train\_input
diskrim.add(Dense(units=400)) #Mensetting Dense
diskrim.add(Dense(units=200, activation='sigmoid')) #Mensett
diskrim.compile(loss='binary\_crossentropy', optimizer=adam\
diskrim.summary()#Memproses data yang sudah disetting dan me
```

Pada contoh tersebut, data akan diambil dari hasil proses sebelumnya yaitu proses ekstrasi data gambar. Dari contoh ini, terdapat 3 layer dense, 1 input layer dan 1 output layer. Pada proses ini, seluruh data akan dibandingkan dengan data sebelumnya yaitu dari generator dan dari data aslinya yang sudah dijadikan data vector.

Gambar 8.83 Teori No 10

11. Jelaskan bagaimana kaitan output dan input antara generator dan diskriminator tersebut. Jelaskan kenapa inputan dan outputan seperti itu

    Pada kedua metode tersebut, akan disebutkan berapa akurasi dari setiap metode. Pada setiap metode tersebut (Discriminator dan generator) akan dilakukan pelatihan dan akan dibandingkan hasilnya. Generator akan menghasilkan data baru sesuai dengan hasil latihan dan dari data tersebut, discriminator akan membandingkan dengan data set apakah data tersebut ”asli” atau tidak.

12. Jelaskan apa perbedaan antara Kullback-Leibler divergence (KL divergence)/relative entropy, Jensen-Shannon(JS) divergence / information radius(iRaD) / total divergence to the average dalam mengukur kualitas dari model

relative entropy adalah ukuran dari bagaimana satu distribusi probabilitas berbeda dari yang kedua, distribusi probabilitas referensi, Divergensi Jensen-Shannon adalah ukuran divergensi berprinsip yang selalu terbatas untuk variabel acak terbatas.

13. Jelaskan apa itu fungsi objektif yang berfungsi untuk mengukur kesamaan antara gambar yang dibuat dengan yang asli.

    Fungsi objektif adalah fungsi yang digunakan sebagai penujuk berapa nilai kesamaan anatara gambar yang dibuat dengan yang asli

14. Jelaskan apa itu scoring algoritma selain mean square error atau cross entropy seperti The Inception Score dan The Frechet Inception distance.

    Inception Score digunakan untuk mengukur seberapa realistis output dari GAN, dimana ada dua parameter, yaitu : gambarnya punya variasi dan setiap gambar jelas terlihat seperti sesuatu. Frechet Inception Distance adalah ukuran kesamaan antara dua dataset gambar. Itu terbukti berkorelasi baik dengan penilaian manusia terhadap kualitas visual dan paling sering digunakan untuk mengevaluasi kualitas sampel Generative Adversarial Networks. FID dihitung dengan menghitung jarak Frchet antara dua Gaussians dipasang ke representasi fitur dari jaringan Inception.

15. Jelaskan kelebihan dan kekurangan GAN

    - Kelebihan
        (a) GAN Menghasilkan data baru yang bisa hampir mirip dengan data asli. Karena hasil pelatihannya, GAN dapat menghasilkan data gambar, teks, audio, dan video yang dapat dibilang hampir mirip dengan yang aslinya. Berkat hal tersebut, GAN dapat digunakan dalam sistem marketing, e-commerce, games,iklan, dan industri lainnya
        (b) GAN mempelajari representasi data secara internal sehingga beberapa masalah pada machine learning dapat diatasi dengan mudah
        (c) Discriminator yang sudah dilatih dapat menjadi sebuah classifier atau pendeteksi jika data sudah sesuai. Karena Discriminator yang akan menjadi tidak efisien berkat seringnya dilatih
        (d) GAN dapat dilatih menggunakan data yang belum dilabeled
    - Kekurangan
        (a) Data saat diproses oleh metode gan tidak konvergensi
        (b) Jenis sampel yang dihasilkan oleh generator terbatas karena modenya terbatas
        (c) Ketidak seimbangnya antara generator dan discriminator dapat menyebabkan overfitting atau terlalu dekat dengan hasil sampel
        (d) Sangat sensitif dengan data yang sudah diinisiasi sebelumnya

8.5.2 Praktek

1. Jelaskan apa itu 3D convolutions Konvolusi 3D. Konvolusi 3D menerapkan filter 3 dimensi ke dataset dan filter bergerak 3 arah (x, y, z) untuk menghitung representasi fitur level rendah. Bentuk output mereka adalah ruang volume 3 dimensi seperti kubus atau kuboid.

2. Jelaskan dengan kode program arsitektur dari generator networknya, beserta penjelasan input dan output dari generator network.

```
def build_discriminator():
    """
    Create a Discriminator Model using hyperparameters values
    defined as follows
    """

    dis_input_shape = (64, 64, 64, 1)
    dis_filters = [64, 128, 256, 512, 1]
    dis_kernel_sizes = [4, 4, 4, 4, 4]
    dis_strides = [2, 2, 2, 2, 1]
    dis_paddings = ['same', 'same', 'same', 'same', 'valid']
    dis_alphas = [0.2, 0.2, 0.2, 0.2, 0.2]
    dis_activations = ['leaky_relu', 'leaky_relu', 'leaky_relu',
                       'leaky_relu', 'sigmoid']
    dis_convolutional_blocks = 5

    dis_input_layer = Input(shape=dis_input_shape)

    # The first 3D Convolutional block
    a = Conv3D(filters=dis_filters[0],
               kernel_size=dis_kernel_sizes[0],
               strides=dis_strides[0],
               padding=dis_paddings[0])(dis_input_layer)
    # a = BatchNormalization()(a, training=True)
    a = LeakyReLU(dis_alphas[0])(a)

    # Next 4 3D Convolutional Blocks
    for i in range(dis_convolutional_blocks - 1):
        a = Conv3D(filters=dis_filters[i + 1],
                   kernel_size=dis_kernel_sizes[i + 1],
                   strides=dis_strides[i + 1],
                   padding=dis_paddings[i + 1])(a)
        a = BatchNormalization()(a, training=True)
        if dis_activations[i + 1] == 'leaky_relu':
            a = LeakyReLU(dis_alphas[i + 1])(a)
        elif dis_activations[i + 1] == 'sigmoid':
            a = Activation(activation='sigmoid')(a)

    dis_model = Model(inputs=[dis_input_layer], outputs=[a])
    return dis_model
```

generator ialah g_loss, Bentuk jaringan Generator dapat dilihat berkebalikan dengan struktur jaringan saraf pada umumnya. Jaringan Generator menerima input sebuah vektor angka z, kemudian mengubahnya menjadi output gambar tiga dimensi.

3. Jelaskan dengan kode program arsitektur dari diskriminator network, beserta penjelasan input dan outputnya.

```
def build_generator():
    """
    Create a Generator Model with hyperparameters values defined as follows
    """
    z_size = 200
    gen_filters = [512, 256, 128, 64, 1]
    gen_kernel_sizes = [4, 4, 4, 4, 4]
    gen_strides = [1, 2, 2, 2, 2]
    gen_input_shape = (1, 1, 1, z_size)
    gen_activations = ['relu', 'relu', 'relu', 'relu', 'sigmoid']
    gen_convolutional_blocks = 5

    input_layer = Input(shape=gen_input_shape)

    # First 3D transpose convolution(or 3D deconvolution) block
    a = Deconv3D(filters=gen_filters[0],
                 kernel_size=gen_kernel_sizes[0],
                 strides=gen_strides[0])(input_layer)
    a = BatchNormalization()(a, training=True)
    a = Activation(activation='relu')(a)

    # Next 4 3D transpose convolution(or 3D deconvolution) blocks
    for i in range(gen_convolutional_blocks - 1):
        a = Deconv3D(filters=gen_filters[i + 1],
                     kernel_size=gen_kernel_sizes[i + 1],
                     strides=gen_strides[i + 1], padding='same')(a)
        a = BatchNormalization()(a, training=True)
        a = Activation(activation=gen_activations[i + 1])(a)

    gen_model = Model(inputs=[input_layer], outputs=[a])
    return gen_model
```

diskrimanator adalah d_loss, Jaringan Discriminator merupakan jaringan klasifikasi biner yang menerima input gambar tiga dimensi dan mengeluarkan klasifikasi menyatakan input gambar adalah gambar asli dari dataset atau merupakan gambar buatan Generator.

4. Jelaskan proses training 3D-GANs proses training 3D gan yaitu dengan melakukan epoch sebanyak yang ditentukan.

5. Jelaskan bagaimana melakukan settingan awal chapter 02 untuk memenuhi semua kebutuhan sebelum melanjutkan ke tahapan persiapan data. 1. clone github 2. download dataset 3. buat folder baru logs dan results

6. Jelaskan tentang dataset yang digunakan, dari mulai tempat unduh, cara membuka dan melihat data. Sampai deskripsi dari isi dataset dengan detail penjelasan setiap folder/file yang membuat orang awam paham. dataset digunakan yaitu 3DShapeNets yang berisi model model bentuk benda dll, folder train

berisi train dan folder test berisi data testing. dan semua data tersebut di simpan didalam folder volumetric_data

7. Jelaskan apa itu voxel dengan ilustrasi dan bahasa paling awam Volume pixel atau voxel adalah titik dalam ruang tiga dimensi. Sebuah voxel mendefinisikan posisi dengan tiga koordinat dalam arah x, y, dan z. Sebuah voxel adalah unit dasar untuk mewakili gambar 3D

8. Visualisasikan dataset tersebut dalam tampilan visual plot, jelaskan cara melakukan visualisasinya 1. import library 2. load data file .mat 3. lalu read memakai matplotlib

9. buka file run.py jelaskan perbaris kode pada fungsi untuk membuat generator yaitu build generator. membuat generator yaitu dengan ketentukan gen sebagai variabel dan membuat fungsi atau variabel gen_model lalu dilakukan return

10. jelaskan juga fungsi untuk membangun diskriminator pada fungsi build discriminator. membangun diskriminator berfungsi untuk mendefenisikan seluruh gambar yang sudah di load generator sebagai gambar fake dan real

11. Jelaskan kode program

```
if __name__ == '__main__':
```

Jika interpreter python menjalankan if name main sebagai program utama, itu ialah menetapkan variabel name untuk memiliki nilai main. Jika file ini sedang diimpor dari modul lain, name akan ditetapkan ke nama modul. Nama modul tersedia sebagai nilai untuk name variabel global.

12. Jelaskan kode program

```
    """
    Specify Hyperparameters
    """
    object_name = "chair"
    data_dir = "3DShapeNets/volumetric_data/" \
               "{}/30/train/*.mat".format(object_name)
    gen_learning_rate = 0.0025
    dis_learning_rate = 10e-5
    beta = 0.5
    batch_size = 1
    z_size = 200
    epochs = 10
    MODE = "train"
```

artinya adalah load dataset yang hanya dalam folder chair data train

13. Jelaskan kode program

```
    """
    Create models
    """
    gen_optimizer = Adam(lr=gen_learning_rate, beta_1=beta)
```

```
    dis_optimizer = Adam(lr=dis_learning_rate, beta_1=beta)

    discriminator = build_discriminator()
    discriminator.compile(loss='binary_crossentropy', optimizer=
    dis_optimizer)

    generator = build_generator()
    generator.compile(loss='binary_crossentropy', optimizer=
    gen_optimizer)
```

disini menggunakan Adam sebagai algoritma pengoptimalan dan binary_crossentropy sebagai kerugian loss.

14. Jelaskan kode program

```
    input_layer = Input(shape=(1, 1, 1, z_size))
    generated_volumes = generator(input_layer)
    validity = discriminator(generated_volumes)
    adversarial_model = Model(inputs=[input_layer], outputs=[
    validity])
    adversarial_model.compile(loss='binary_crossentropy',
    optimizer=gen_optimizer)
```

ini artinya ialah kita memasukkan random vector kedalam generator model lalu membagi 2 yaitu generated example dan real example, dan meneruskan ke diskriminator model sebagai real atau fake

15. Jelaskan kode program

```
    print("Loading data...")
    volumes = get3DImages(data_dir=data_dir)
    volumes = volumes[..., np.newaxis].astype(np.float)
    print("Data loaded...")
```

ini melakukan load data pada dataset

16. Jelaskan kode program

```
  tensorboard = TensorBoard(log_dir="logs/{}".format(time.time())
    )
    tensorboard.set_model(generator)
    tensorboard.set_model(discriminator)
```

ini berfungsi untuk membuat tensorboard yang nantinya bisa diakses melalui localhost

17. Jelaskan kode program

```
    labels_real = np.reshape(np.ones((batch_size,)), (-1, 1, 1,
    1, 1))
    labels_fake = np.reshape(np.zeros((batch_size,)), (-1, 1, 1,
    1, 1))
```

fungsi ini ialah untuk melakukan reshape agar shape yang dihasilkan tidak terlalu besar.

18. Jelaskan kode program

```
    if MODE == 'train':
        for epoch in range(epochs):
            print("Epoch:", epoch)

            gen_losses = []
            dis_losses = []
```

karena jika epoch semakin banyak maka kualiatas training yang dihasilkan akan semakin baik

19. Jelaskan kode program

```
            number_of_batches = int(volumes.shape[0] / batch_size)
            print("Number of batches:", number_of_batches)
            for index in range(number_of_batches):
                print("Batch:", index + 1)
```

batch adalah jumlah file yang akan di training

20. Jelaskan kode program

```
                z_sample = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                volumes_batch = volumes[index * batch_size:(index + 1) * batch_size, :, :, :]
```

ini adalah untuk membuat gambar bersih dari noise dan juga menyesuaikan shape

21. Jelaskan kode program

```
                # Next, generate volumes using the generate network
                gen_volumes = generator.predict_on_batch(z_sample)
```

ialah membuat sample gambar palsu yang akan diteruskan ke diskriminator

22. Jelaskan kode program

```
                """
                Train the discriminator network
                """
                discriminator.trainable = True
                if index % 2 == 0:
                    loss_real = discriminator.train_on_batch(volumes_batch, labels_real)
                    loss_fake = discriminator.train_on_batch(gen_volumes, labels_fake)

                    d_loss = 0.5 * np.add(loss_real, loss_fake)
                    print("d_loss:{}".format(d_loss))

                else:
                    d_loss = 0.0
```

diskriminator bisa load gambar fake dan real dari generator, oleh karena itu ada generator loss dan diskriminator loss untuk melihat seberapa baik kualitas yang dihasilkan

23. Jelaskan kode program

```
                z = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                g_loss = adversarial_model.train_on_batch(z, labels_real)
                print("g_loss:{}".format(g_loss))

                gen_losses.append(g_loss)
                dis_losses.append(d_loss)
```

dengan melakukan print g_loss untuk generator dan juga d_loss untuk diskriminator

24. Jelaskan kode program

```
                # Every 10th mini-batch, generate volumes and save them
                if index % 10 == 0:
                    z_sample2 = np.random.normal(0, 0.33, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
                    generated_volumes = generator.predict(z_sample2, verbose=3)
                    for i, generated_volume in enumerate(generated_volumes[:5]):
                        voxels = np.squeeze(generated_volume)
                        voxels[voxels < 0.5] = 0.
                        voxels[voxels >= 0.5] = 1.
                        saveFromVoxels(voxels, "results/img_{}_{}_{}".format(epoch, index, i))
```

mengapa ada perulangan karena untuk melakukan perbandingan dari hasil yang sudah didapat.

25. Jelaskan kode program

```
            # Write losses to Tensorboard
            write_log(tensorboard, 'g_loss', np.mean(gen_losses), epoch)
            write_log(tensorboard, 'd_loss', np.mean(dis_losses), epoch)
```

TensorBoard adalah sebuah aplikasi web localhost untuk memeriksa dan menyelesaikan grafik dari hasil TensorFlow.

26. Jelaskan kode program

```
        """
        Save models
        """
        generator.save_weights(os.path.join("models", "generator_weights.h5"))
```

```
        discriminator.save_weights(os.path.join("models", "discriminator_weights.h5"))
```

File H5 adalah file data yang disimpan dalam Format Data Hirarki (HDF). Ini berisi array multidimensi data ilmiah

27. Jelaskan kode program

```
    if MODE == 'predict':
        # Create models
        generator = build_generator()
        discriminator = build_discriminator()

        # Load model weights
        generator.load_weights(os.path.join("models", "generator_weights.h5"), True)
        discriminator.load_weights(os.path.join("models", "discriminator_weights.h5"), True)

        # Generate 3D models
        z_sample = np.random.normal(0, 1, size=[batch_size, 1, 1, 1, z_size]).astype(np.float32)
        generated_volumes = generator.predict(z_sample, verbose=3)

        for i, generated_volume in enumerate(generated_volumes[:2]):
            voxels = np.squeeze(generated_volume)
            voxels[voxels < 0.5] = 0.
            voxels[voxels >= 0.5] = 1.
            saveFromVoxels(voxels, "results/gen_{}".format(i))
```

ini adalah tahap akhir untuk melakukan testing dari model yang telah dibuat dan buat model dari yang sudah di create sebelumnya yaitu generator dan diskriminator.

```
    ...:             saveFromVoxels(voxels,
Loading data...
Data loaded...
Epoch: 0
Number of batches: 10
Batch: 1
d_loss:0.6931722164154053
g_loss:0.6931138038635254
Batch: 2
g_loss:0.6931138038635254
Batch: 3
d_loss:0.6931579113006592
g_loss:0.6931090354919434
Batch: 4
g_loss:0.6931090354919434
Batch: 5
d_loss:0.6931560039520264
g_loss:0.693108081817627
```

Gambar 8.84 Result

Gambar 8.85 Result

Gambar 8.86 Result

Gambar 8.87 Result

BAB 9

CHAPTER 9

9.1 1174086 - Tia Nur Candida

9.1.1 Teori

1. Jelaskan dengan ilustrasi gambar sendiri apa perbedaan antara vanilla GAN dan cGAN.
   Vanilla GANs biasanya tidak memiliki convolutional Neural Jaringan (CNNs) di jaringan mereka. Conditional GANs (cGANs) adalah perpanjangan dari model GAN. Mereka memungkinkan untuk generasi gambar yang memiliki kondisi tertentu atau atribut dan telah terbukti menjadi lebih baik dari Vanilla GANs sebagai hasilnya.
   cGANs adalah jenis GAN yang dikondisikan pada beberapa informasi tambahan. informasi tambahan y ke Generator sebagai lapisan input tambahan. Dalam Vanilla GANs, tidak ada kontrol atas Kategori gambar yang dihasilkan. Ketika kita menambahkan kondisi y ke Generator, kita dapat menghasilkan gambar dari kategori tertentu, menggunakan y, yang mungkin jenis data, seperti label kelas atau data integer. Vanilla GANs bisa belajar hanya satu kategori dan sangat sulit untuk arsitek GANs untuk beberapa kategori. Sebuah cGAN,

By Rolly M. Awangga Copyright © 2020 Kreatif Industri Nusantara.

bagaimanapun, dapat digunakan untuk menghasilkan model multi-modal dengan kondisi yang berbeda untuk kategori yang berbeda.

Gambar 9.1 Illustrasi Vanilla GAN dan cGAN

2. Jelaskan dengan ilustrasi gambar sendiri arsitektur dari Age-cGAN.

   Arsitektur cGAN untuk penuaan wajah sedikit lebih rumit. AgecGan terdiri dari empat jaringan: Encoder, FaceNet, Jaringan Generator, dan jaringan diskriminator. Dengan Encoder, kita belajar pemetaan invers gambar wajah masukan dan kondisi usia dengan vektor laten. FaceNet adalah jaringan pengenalan wajah yang mempelajari perbedaan antara gambar input x dan gambar yang direkonstruksi. Kami memiliki jaringan Generator, yang mengambil representasi tersembunyi yang terdiri dari gambar wajah dan vektor kondisi dan menghasilkan gambar. Jaringan diskriminator adalah untuk mendiskriminasikan antara gambar nyata dan gambar palsu. Masalah dengan cGANs adalah bahwa mereka tidak dapat mempelajari tugas pemetaan terbalik masukan gambar x dengan atribut y ke vektor laten z. Solusi untuk masalah ini adalah dengan menggunakan jaringan Encoder. Kita dapat melatih jaringan encoder untuk memperkirakan pemetaan terbalik dari input Images x.

Gambar 9.2 Illustrasi Arsitektur cGAN

3. Jelaskan dengan ilustrasi gambar sendiri arsitektur encoder network dari AgecGAN.
   Tujuan utama dari jaringan Encoder adalah untuk menghasilkan vektor laten dari gambar yang disediakan. Pada dasarnya, dibutuhkan gambar dimensi (64,

64, 3) dan mengubahnya menjadi vektor 100-dimensi. Jaringan Encoder adalah jaringan syaraf convolutional yang dalam. Jaringan berisi empat convolutional blok dan dua lapisan padat. Setiap blok convolutional berisi lapisan convolutional, lapisan normalisasi batch, dan fungsi aktivasi. Di setiap blok convolutional, setiap lapisan convolutional diikuti oleh lapisan normalisasi batch, kecuali lapisan convolutional pertama.

Gambar 9.3 Illustrasi Network Encoder

4. Jelaskan dengan ilustrasi gambar sendiri arsitektur generator network dari Agec-GAN.
   Tujuan utama dari generator adalah untuk menghasilkan gambar dari dimensi (64, 64, 3). Dibutuhkan vektor laten 100 dimensi dan beberapa informasi tambahan, y, dan mencoba untuk menghasilkan gambar yang realistis. Jaringan Generator adalah jaringan neural yang mendalam convolutional juga. Hal ini terdiri dari lapisan padat, upsampling, dan convolutional. Dibutuhkan dua nilai input: vektor kebisingan dan nilai pengkondisian. Nilai pengkondisian adalah informasi tambahan yang diberikan ke jaringan. Untuk Age-cGAN, ini akan menjadi usia.

Gambar 9.4 Illustrasi Network Generator

Jelaskan dengan ilustrasi gambar sendiri arsitektur discriminator network dari Age-cGAN.

Tujuan utama dari jaringan diskriminator adalah untuk mengidentifikasi apakah gambar yang disediakan adalah palsu atau nyata. Hal ini dilakukan dengan melewati gambar melalui serangkaian lapisan sampling bawah dan beberapa lapisan klasifikasi. Dengan kata lain, ini memprediksi Apakah gambar itu nyata atau palsu. Seperti jaringan lain, Jaringan diskriminator lain dalam jaringan convolutional. Ini berisi beberapa blok convolutional. Setiap blok convolutional berisi lapisan convolutional, lapisan normalisasi batch, dan fungsi aktivasi, selain blok convolutional pertama, yang tidak memiliki lapisan normalisasi batch.

Gambar 9.5 Illustrasi Discriminator Network

5. Jelaskan dengan ilustrasi gambar apa itu pretrained Inception-ResNet-2 Model. pre-trained Inception-ResNet-2 network, sekali disediakan dengan gambar, mengembalikan yang sesuai embedding. Tertanam yang diekstrak untuk gambar asli dan gambar direkonstruksi dapat dihitung dengan menghitung jarak Euclidean dari yang tertanam.

Gambar 9.6 Illustrasi Inception-ResNet-2 Model.

6. Jelaskan dengan ilustrasi gambar sendiri arsitektur Face recognition network Age-cGAN.

   Tujuan utama dari jaringan pengenalan wajah adalah untuk mengenali identitas seseorang dalam gambar yang diberikan.

Gambar 9.7 Illustrasi Face recognition network Age-cGAN.

7. . Sebutkan dan jelaskan serta di sertai contoh-contoh tahapan dari Age-cGAN

   Age-cGAN memiliki beberapa tahapan pelatihan. Seperti disebutkan di bagian

sebelumnya, Age-cGAN memiliki empat jaringan, yang dilatih dalam tiga tahap. Pelatihan AgecGAN terdiri dari tiga tahap:

- pelatihan GAN bersyarat: pada tahap ini, kita melatih jaringan Generator dan jaringan diskriminator.
- awal pendekatan vektor laten: pada tahap ini, kami melatih jaringan Encoder.
- optimasi vektor laten: pada tahap ini, kami mengoptimalkan kedua encoder dan jaringan generator.

8. Berikan contoh perhitungan fungsi training objektif

9. Berikan contoh dengan ilustrasi penjelasan dari Initial latent vector approximation
   Perkiraan vektor laten awal adalah metode untuk memperkirakan vektor laten untuk mengoptimalkan rekonstruksi gambar wajah. Untuk memperkirakan vektor laten, kami memiliki jaringan Encoder. Kami melatih jaringan Encoder pada gambar yang dihasilkan dan gambar nyata. Setelah dilatih, Jaringan Encoder akan mulai menghasilkan vektor laten dari Distribusi. Tujuan pelatihan fungsi untuk pelatihan jaringan Encoder adalah kehilangan jarak Euclidean.

Gambar 9.8 Illustrasi Initial latent vector approximation

10. Berikan contoh perhitungan latent vector optimization

Gambar 9.9 Contoh Perhitungan Latent vector optimization

9.1.2 Praktek

1. Jelaskan bagaimana cara ekstrak le dataset Age-cGAN menggunakan google colab. Menggunakan Google Colab, dimana membuat notebooks baru, kemudian membuat ekstraksi file dari link dataset.

```
# In[1. Ekstrak File]:
import tarfile
```

```
tf = tarfile.open("/content/drive/My Drive/Colab Notebooks/
    wiki_crop.tar")
tf.extractall(path="/content/drive/My Drive/Colab Notebooks")
```

2. Jelaskan bagaimana kode program bekerja untuk melakukan load terhadap dataset yang sudah di ekstrak, termasuk bagaimana penjelasan kode program perhitungan usia. Dibawah ini merupakan code untuk melakukan fungsi perhitungan usia.

```
# In[2. Load Data]:
def load_data(wiki_dir, dataset='wiki'):
    # Load the wiki.mat file
    meta = loadmat(os.path.join(wiki_dir, "{}.mat".format(dataset
    )))

    # Load the list of all files
    full_path = meta[dataset][0, 0]["full_path"][0]

    # List of Matlab serial date numbers
    dob = meta[dataset][0, 0]["dob"][0]

    # List of years when photo was taken
    photo_taken = meta[dataset][0, 0]["photo_taken"][0]  # year

    # Calculate age for all dobs
    age = [calculate_age(photo_taken[i], dob[i]) for i in range(
    len(dob))]

    # Create a list of tuples containing a pair of an image path
    and age
    images = []
    age_list = []
    for index, image_path in enumerate(full_path):
        images.append(image_path[0])
        age_list.append(age[index])

    # Return a list of all images and respective age
    return images, age_list
```

3. Jelaskan bagaimana kode program The Encoder Network bekerja dijelaskan dengan bahawa awam dengan ilustrasi sederhana. Proses Encoder berfungsi untuk mempelajari pemetaan terbalik dari gambar wajah dan kondisi usia dengan vector latent Z.

```
# In[3. Encoder Bekerja]:
def build_encoder():
    """
    Encoder Network
    """
    input_layer = Input(shape=(64, 64, 3))

    # 1st Convolutional Block
    enc = Conv2D(filters=32, kernel_size=5, strides=2, padding='
    same')(input_layer)
```

```
    # enc = BatchNormalization()(enc)
    enc = LeakyReLU(alpha=0.2)(enc)

    # 2nd Convolutional Block
    enc = Conv2D(filters=64, kernel_size=5, strides=2, padding='
    same')(enc)
    enc = BatchNormalization()(enc)
    enc = LeakyReLU(alpha=0.2)(enc)

    # 3rd Convolutional Block
    enc = Conv2D(filters=128, kernel_size=5, strides=2, padding='
    same')(enc)
    enc = BatchNormalization()(enc)
    enc = LeakyReLU(alpha=0.2)(enc)

    # 4th Convolutional Block
    enc = Conv2D(filters=256, kernel_size=5, strides=2, padding='
    same')(enc)
    enc = BatchNormalization()(enc)
    enc = LeakyReLU(alpha=0.2)(enc)

    # Flatten layer
    enc = Flatten()(enc)

    # 1st Fully Connected Layer
    enc = Dense(4096)(enc)
    enc = BatchNormalization()(enc)
    enc = LeakyReLU(alpha=0.2)(enc)

    # Second Fully Connected Layer
    enc = Dense(100)(enc)

    # Create a model
    model = Model(inputs=[input_layer], outputs=[enc])
    return model
```

4. Jelaskan bagaimana kode program The Generator Network bekerja dijelaskan dengan bahawa awam dengan ilustrasi sederhana. Proses Generator agar bekerja dengan baik dibutuhkan representasi dari gambar wajah dan vector kondisi sebagai inputan yang menghasilkan sebuah gambar.

```
# In[4. Generator Network Bekerja]:
def build_generator():
    """
    Create a Generator Model with hyperparameters values defined
    as follows
    """
    latent_dims = 100
    num_classes = 6

    input_z_noise = Input(shape=(latent_dims,))
    input_label = Input(shape=(num_classes,))

    x = concatenate([input_z_noise, input_label])

    x = Dense(2048, input_dim=latent_dims + num_classes)(x)
```

```
    x = LeakyReLU(alpha=0.2)(x)
    x = Dropout(0.2)(x)

    x = Dense(256 * 8 * 8)(x)
    x = BatchNormalization()(x)
    x = LeakyReLU(alpha=0.2)(x)
    x = Dropout(0.2)(x)

    x = Reshape((8, 8, 256))(x)

    x = UpSampling2D(size=(2, 2))(x)
    x = Conv2D(filters=128, kernel_size=5, padding='same')(x)
    x = BatchNormalization(momentum=0.8)(x)
    x = LeakyReLU(alpha=0.2)(x)

    x = UpSampling2D(size=(2, 2))(x)
```

5. Jelaskan bagaimana kode program The Discriminator Network bekerja dijelaskan dengan bahawa awam dengan ilustrasi sederhana. Proses Discriminator untuk membedakan antara gambar asli dan gambar palsu.

```
# In[5.Discriminator Network Bekerja]:
def build_discriminator():
    """
    Create a Discriminator Model with hyperparameters values
    defined as follows
    """
    input_shape = (64, 64, 3)
    label_shape = (6,)
    image_input = Input(shape=input_shape)
    label_input = Input(shape=label_shape)

    x = Conv2D(64, kernel_size=3, strides=2, padding='same')(
    image_input)
    x = LeakyReLU(alpha=0.2)(x)

    label_input1 = Lambda(expand_label_input)(label_input)
    x = concatenate([x, label_input1], axis=3)

    x = Conv2D(128, kernel_size=3, strides=2, padding='same')(x)
    x = BatchNormalization()(x)
    x = LeakyReLU(alpha=0.2)(x)

    x = Conv2D(256, kernel_size=3, strides=2, padding='same')(x)
    x = BatchNormalization()(x)
    x = LeakyReLU(alpha=0.2)(x)

    x = Conv2D(512, kernel_size=3, strides=2, padding='same')(x)
    x = BatchNormalization()(x)
    x = LeakyReLU(alpha=0.2)(x)

    x = Flatten()(x)
    x = Dense(1, activation='sigmoid')(x)

    model = Model(inputs=[image_input, label_input], outputs=[x])
    return model
```

6. Jelaskan bagaimana kode program Training cGAN bekerja dijelaskan dengan bahawa awam dengan ilustrasi sederhana. Proses Training cGAN ini dengan load file .mat pada dataset lalu epoch sebanuak 500 kali.

```
# In[6. Training cGAN]:
    if __name__ == '__main__':
    # Define hyperparameters
    data_dir = "data"
    wiki_dir = os.path.join(data_dir, "wiki_crop1")
    epochs = 500
    batch_size = 2
    image_shape = (64, 64, 3)
    z_shape = 100
    TRAIN_GAN = True
    TRAIN_ENCODER = False
    TRAIN_GAN_WITH_FR = False
    fr_image_shape = (192, 192, 3)

    # Define optimizers
    dis_optimizer = Adam(lr=0.0002, beta_1=0.5, beta_2=0.999, epsilon=10e-8)
    gen_optimizer = Adam(lr=0.0002, beta_1=0.5, beta_2=0.999, epsilon=10e-8)
    adversarial_optimizer = Adam(lr=0.0002, beta_1=0.5, beta_2=0.999, epsilon=10e-8)
```

7. Jelaskan bagaimana kode program Initial dan latent vector approximation bekerja dijelaskan dengan bahawa awam dengan ilustrasi sederhana. Initial dan Latent Vector Approximation bekerja melakukan predicsi epoch yang telah di buat sebanyak 500 kali, dan nanti hasilnya ada di folder result.

```
# In[7. Laten Vector]:
    """
    Train encoder
    """

    if TRAIN_ENCODER:
        # Build and compile encoder
        encoder = build_encoder()
        encoder.compile(loss=euclidean_distance_loss, optimizer='adam')

        # Load the generator network's weights
        try:
            generator.load_weights("generator.h5")
        except Exception as e:
            print("Error:", e)

        z_i = np.random.normal(0, 1, size=(5000, z_shape))

        y = np.random.randint(low=0, high=6, size=(5000,), dtype=np.int64)
        num_classes = len(set(y))
        y = np.reshape(np.array(y), [len(y), 1])
        y = to_categorical(y, num_classes=num_classes)
```

```
23
24      for epoch in range(epochs):
25          print("Epoch:", epoch)
26
27          encoder_losses = []
28
29          number_of_batches = int(z_i.shape[0] / batch_size)
30          print("Number of batches:", number_of_batches)
31          for index in range(number_of_batches):
32              print("Batch:", index + 1)
33
34              z_batch = z_i[index * batch_size:(index + 1) *
    batch_size]
35              y_batch = y[index * batch_size:(index + 1) *
    batch_size]
36
37              generated_images = generator.predict_on_batch([
    z_batch, y_batch])
38
39              # Train the encoder model
40              encoder_loss = encoder.train_on_batch(
    generated_images, z_batch)
41              print("Encoder loss:", encoder_loss)
42
43              encoder_losses.append(encoder_loss)
44
45          # Write the encoder loss to Tensorboard
46          write_log(tensorboard, "encoder_loss", np.mean(
    encoder_losses), epoch)
47
48      # Save the encoder model
49      encoder.save_weights("encoder.h5")
```

9.1.3 Bukti Tidak Plagiat

SmallSEOTools

PLAGIARISM SCAN REPORT

Words	788	Date	June 13,2020
Characters	5936	Exclude Url	

0% Plagiarism	100% Unique	0 Plagiarized Sentences	51 Unique Sentences

Content Checked For Plagiarism

No 1 Vanilla GANs biasanya tidak memiliki convolutional Neural Jaringan (CNNs) di jaringan mereka. Conditional GANs (cGANs) adalah perpanjangan dari model GAN. Mereka memungkinkan untuk generasi gambar yang memiliki kondisi tertentu atau atribut dan telah terbukti menjadi lebih baik dari Vanilla GANs sebagai hasilnya. Seperti ini, kami telah menerapkan jaringan GAN yang berbeda untuk kasus penggunaan yang berbeda. Dilain banyaknya memperhaus Ide Vanilla GANs dan memungkinkan kita untuk mengontrol output dari jaringan generator. Perusaan wajah adalah tentang mengubah usia wajah seseorang tanpa mengubah identitas. Dalam kebanyakan model lain (termasuk GANs), penampilan atau identitas seseorang hilang oleh GAN karena ekspresi wajah dan aksesoris wajah, seperti kacamata hitam atau jenggot, tidak dipertahankan. Age-cGANs memperlimbangkan semua atribut ini. Pada bagian ini, kita akan menjelajahi cGANs untuk menghadapi penuaan cGANs adalah jenis GAN yang dikondisikan pada beberapa informasi tambahan. Kami menberi makan informasi tambahan y ke Generator sebagai lapisan input tambahan. Dalam Vanilla GANs, tidak ada kontrol atas Kategori gambar yang dihasilkan. Dengan cGANs, kita dapat memilih kelas yang ingin kita hasilkan gambar untuk. Dataset MNIST adalah dataset yang mengandung yang mungkin jenis data, seperti label kelas atau data integer. Vanilla GANs bisa belajar hanya satu kategori dan sangat sulit untuk ardele GANs untuk beberapa kategori. Sebuah cGAN, bagaimanapun, dapat digunakan untuk menghasilkan model multi-modal dengan kondisi yang berbeda untuk kategori yang berbeda. No 2 Arsitektur cGAN untuk penuaan wajah sedikit lebih rumit. AgeGan terdiri dari empat jaringan: Encoder, Facenet, jaringan Generator, dan jaringan diskriminator. Dengan Encoder, kita belajar pemetaan invers gambar wajah masukan dan kondisi usia dengan vektor laten. Facenet adalah jaringan pengenalan wajah yang mempelajari perbedaan antara gambar input x dan gambar yang direkonstruksi. Kami memiliki jaringan Generator, yang mengambil representasi tersembunyi yang terdiri dari gambar wajah dan vektor kondisi dan menghasilkan gambar. Jaringan diskriminator adalah untuk mendiskriminasikan antara gambar nyata dan gambar palsu. Masalah dengan cGANs adalah bahwa mereka tidak dapat mempelajari tugas pemetaan terbalik masukan gambar x dengan atribut y ke vektor laten z. Solusi untuk masalah ini adalah dengan menggunakan jaringan Encoder. Kita dapat melatih jaringan encoder untuk memperkirakan pemetaan terbalik dari input Images o. Pada bagian ini, kita akan menjelajahi jaringan yang terlibat dalam Age-cGAN. No3 Tujuan utama dari jaringan Encoder adalah untuk menghasilkan vektor laten dari gambar yang diberikan. Pada dasarnya, dibutuhkan gambar dimensi 64, 64, 3) dan mengubahnya menjadi vektor 100-dimensi. Jaringan Encoder adalah jaringan syaraf convolutional yang dalam. Jaringan berisi empat convolutional blok dan dua lapisan

Gambar 9.10 Tidak Melakukan Plagiat Pada Ch 9