PEMBELAJARAN MENDALAM: VISI KOMPUTER TINGKAT LANJUT

PEMBELAJARAN MENDALAM: VISI KOMPUTER TINGKAT LANJU
Dalam 24 Jam

Rolly M. Awangga
Informatics Research Center

Kreatif Industri Nusantara

Penulis:
Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:
M. Yusril Helmi Setyawan

Penyunting:
Syafrial Fachrie Pane
Khaera Tunnisa
Diana Asri Wijayanti

Desain sampul dan Tata letak:
Deza Martha Akbar

Penerbit:
Kreatif Industri Nusantara

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

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

Cetakan Pertama, 2019

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

CONTRIBUTORS

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

CONTENTS IN BRIEF

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Listings

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

CNN	Convolutional Neural Network
MNIST	Modified National Institute of Standards and Technology
SSH	Secure Shell
HTTPS	Hypertext Transfer Protocol Secure

GLOSSARY

Sotfware	Adalah suatu bagian dari sistem komputer yang tidak memiliki wujud fisik dan tidak terlihat karena merupakan sekumpulan data elektronik yang disimpan dan diatur oleh komputer berupa program yang dapat menjalankan suatu perintah.
Feedforwad	Adalah suatu struktur jaringan syaraf tiruan yang meimiliki karakteristik dengan tidak adanya pengulangan pembelajaran dimana signal bergerak dari layer input dan melewati layer tersembunyi dan kemudian menuju layer output.
Python	Adalah bahasa pemrograman interpretatif multiguna, python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks.
convolutional	Adalah suatu istilah matematis yang berati mengaplikasikan sebuah fungsi pada output fungsi lain secara berulang.

SYMBOLS

A Amplitude

& Propositional logic symbol

a Filter Coefficient

$\mathcal{B}$ Number of Beats

INTRODUCTION

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

Informatics Research Center
Bandung, Jawa Barat, Indonesia

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

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

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

BAB 1

INTRODUCTION

Halo semua dan selamat datang di kursus Neural Convolutional, dengan pembelajaran python bagian 9. Neural Convolutional adalah salah satu kursus yang menarik dan membuktikan bahwa siswa yang mengikuti kursus ini akan lebih cepat dan mudah mengerti. Saya akan memberikan pembelajaran tentang pendalaman materi yang dapat membantu anda dalam pembelajaran dengan banyaknya materi- materi. Jadi, izinkan saya menjelaskan dengan secara singkat tentang kursus ini :

Dalam kursus ini kita akan mempelajari bagaimana cara mengatur antara arsitektur CNN dasar yang sudah anda kenal dan menikmati arsitektur novel modern seperti Viji Reznick dan Inception yang mungkin akan anda beri sesuai nama film. Kami akan menggunakan ini pada gambar sel sel dan membuat sistem yang lebih baik. Salah satu tema utama dari kursus ini juga beralih dari CNN ke sistem yang melibatkan CNN. CNN juga membuat satu gambar klasifikasi hal dasar dalam kursus ini, anda akan melihat bagaimana kita dapat mengubah CNN menjadi sistem deteksi objek yang tidak hanya mengklasifikasikan gambar tetapi juga dapat menemukan setiap objek dalam gambar dan prediksi labelnya.

1.1 Jaringan saraf convolutional canggih

Tujuan Pembelajaran

1. kita telah melihat bahwa 3-5 layer netscan membutuhkan waktu yang sangat lama untuk dilatih (tapi sekarang kita akan melihat 50 layer nets)
2. penelitian hari ini (dalam pembelajaran mesin) berkomitmen untuk keterbukaan, dan dengan membagikan penelitian mereka, mudah bagi Anda untuk melakukan hal-hal canggih di rumah (Tidak ada bidang lain yang bisa mencapai ini: biologi, kedokteran, fisika, ...dan lain sebagainya)
3. kita dapat menggunakan bobot pra-terlatih menggunakan transfer belajar secara signifikan mengurangi waktu pelatihan karena kita sekarang hanya perlu melakukan fine-tuning

1.2 cara untuk melakukan kursus ini dengan baik

Saya telah menemukan solusi ini setelah mengamati siswa/i selama bertahun-tahun, pada umumnya, mereka yang mengikuti solusi ini telah mendapatkan kesuksesan, mereka yang memiliki masalah disebabkan karena tidak mengikuti solusi ini.

Hal-hal atau solusi yang diperlukan antara lain :

1. memanfaatkan dengan adanya Question and Answer
2. memerlukan waktu respon yang cepat
3. mempunyai motifasi yang tinggi
4. menggunakan sotfware yang kita mengerti dan kita pahami

1.3 Convolutional Neural Network

Ulasan tentang CNN

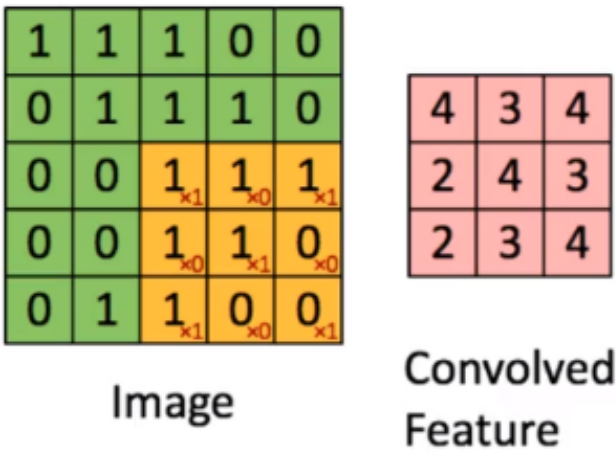
1. Memahami penulisan jaringan saraf feedforward menggunakan beberapa pustaka
2. Mengetahui secara umum bagaimana jaringan saraf bekerja, bagaimana melatihnya pada data, seperti apa data itu (formatnya), bagaimana membuat prediksi baru tentang data tersebut.
3. Mengetahui tentang convolution

Convolution

1. Filter(3X3) adalah tensor berat yang dipelajari dengan backpropagation.

2. Kesalahan : merancang filter untuk menjadi pendeteksi tepi dll.

3. Tidak dapat diskalakan: CNN berisi seribuan filter saraf sehingga tidak mungkin anda dapat memperbaiki dengan cara yang dapat dilakukan sesuai kebutuhan anda



Gambar 1.1 Ilustasi gambar pada setiap pergeseran filter

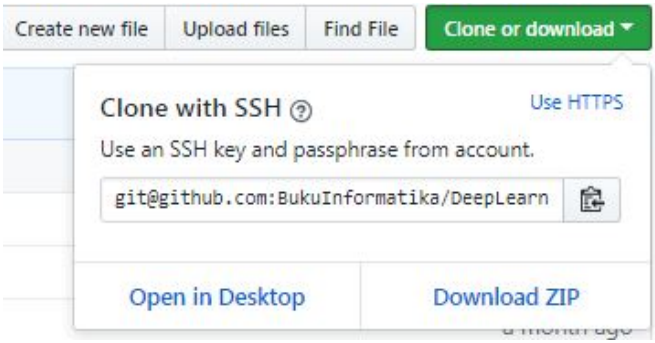
Convolutional Neural Network (CNN) adalah salah satu jenis neural network yang biasa digunakan pada data image. CNN bisa digunakan untuk mendeteksi dan mengenali object pada sebuah image. Secara garis besar CNN tidak jauh beda dengan neural network biasanya. CNN terdiri dari neuron yang memiliki weight, bias dan activation function. Filter pada dasarnya meluncur diatas setiap posisi yang mungkin pada gambar dan pada setiap bagian yang tumpang tindih mendapatkan elemen berlipat ganda untuk pengadaan dan penambahan konvolusi ini merupakan konsep matrix seperti teknik pengukuran jarak ataupun mengukur korelasi sebuah matrix Jika filter sangat berkolerasi dengan potongan gambar maka akan menghasilkan jumlah yang sangat besar dan jika filter sangat berbeda dengan potongan gambar maka akan menghasilkan jumlah yang sangat kecil dalam aktualitas yang disebut konvolusi atau disebut sebagai korelasi silang karena konvolusi merupakan sebuah konsep

1.4 cara untuk mendapatkan kode dan data

Langkahnya sebagai berikut

1. mengambil kode ssh yang ada pada github dan pastikan yang dicopy adalah ssh bukan https

2. jangan lupa melakukan fork terlebih dahulu



Gambar 1.2 contoh pengambilan kode ssh

3. kemudian disarankan untuk tidak memalsukan repo karena dapat mempersulit

Dalam hal data, data akan kita temui pada saat kita dalam kuliah dan kursus, beberapa siswa telah meminta saya untuk memberikan tentang tutorial latihan coding dalam kursus ini jadi tidak ada alasan lagi untuk tidak dapat mengoding

1. Jaringan saraf konvolutional canggih dalam kuliah ini saya akan membahas bagaimana untuk mendapatkan kode untuk kursus ini. Jadi seperti biasa kode dalam kursus ini dapat di unduh dari halaman saya. Untuk mendapatkan kunci tersebut, cobalah untuk mencoba dan menempelkan dari halaman web itu sendiri. Cukup gunakan perintah clone lalu buat semua folder yang relevan untuk kursus ini juga kelas CNN.

2. Tidak memalsukan report karena hal ini mempersulit untuk mendapatkan persetujuan dan saya membuat report yang cukup banyak dan terus menerus sehingga membuat anda tidak terjebak dengan versi lama. Selanjutnya jika anda sudah mengambil salah satu kelas saya dan anda sudah memiliki report ini cukup klik tarik dan anda secara otomatis memiliki kode untuk data dalam kursus ini.

3. Umumnya akan melihat data set yang berbeda, dimana untuk mendapatkan data didua tempat, baik dalam kuliah dan dalam kode. Jadi dalam kursus ini anda tidak perlu mengetik kode berulang kali hanya untuk melihat data set yang berbeda.

1.5 Data Fashion MNIST

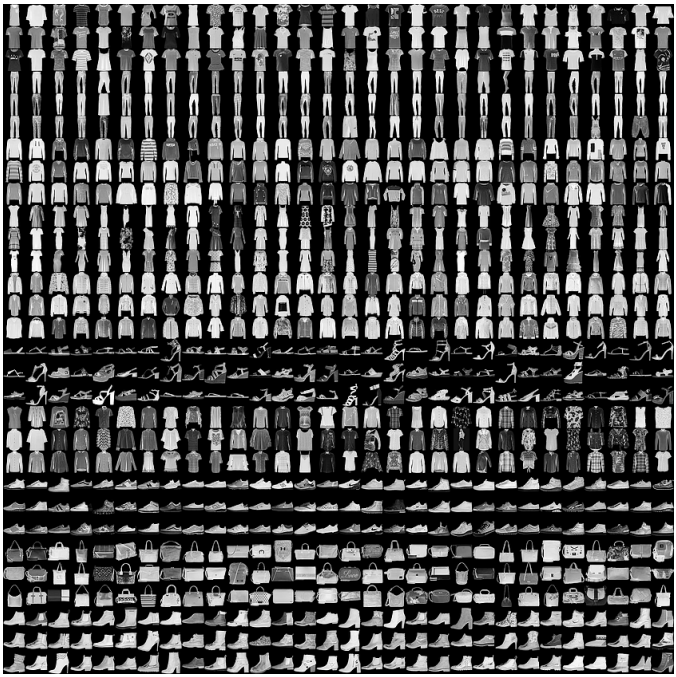
Data Fashion MNIST (Modified National Institute of Standards and Technology) adalah basis data yang berbentuk tulisan angka yang biasa digunakan untuk melatih pola pikir kita dalam algoritma.

Hal hal yang perlu kita lakukan untuk kursus Data Fashion MNIST :

1. Download terlebih dahulu kaggle pada google

2. Sebelum mendownload pastikan kamu telah memiliki akun kaggle
3. Sediakan tempat penyimpanan file yang besar

Kali ini kita akan melakukan contoh klasifikasi terhadap data Fashion MNIST. Fashion MNIST ini adalah dataset yang terdiri dari 10 kategori fashion sebagai berikut :



Gambar 1.3 contoh klarifikasi data Fashion MNIST

Hasil Klarifikasi

1. T-Shirt/Tops = 0
2. Trouser = 1
3. Pullover = 2
4. Dress = 3
5. Coat = 4
6. Sandal = 5
7. Shirt = 6
8. Sneaker = 7

9. Bag = 8
10. Ankle Boot = 9

Tiap kategori terdiri dari 6.000 images untuk training dan 1.000 images untuk testing. Jadi total untuk training data ada 60.000 images dan 10.000 untuk testing data.

1.5.1 Dependency

Dependency yang dibutuhkan pada contoh autoencoder kali ini hampir sama dengan contoh pada part-part sebelumnya. Hanya saja kali ini kita akan load MNIST data dari package yang sudah disediakan oleh Keras. Kita juga mau coba optimizers baru yaitu ADAM.

ADAM adalah variant dari algoritma gradient descent.

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  import tensorflow as tf
4  from keras.models import Model
5  from keras.layers import Input, Activation, Dense
6  from keras.optimizers import Adam
7  from keras.utils.np_utils import to_categorical
8  from keras.datasets import mnist
```

p6_dependency.py hosted with ❤ by GitHub view raw

Gambar 1.4 contoh dependency

1.5.2 Data Preparation

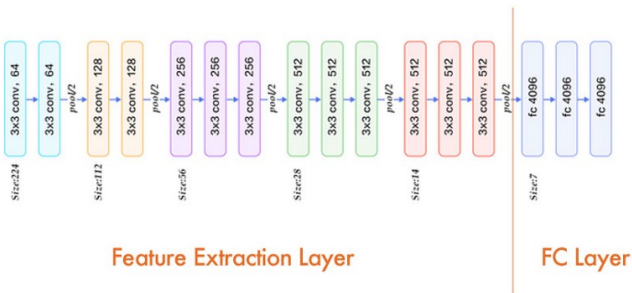
Data dari MNIST ini adalah grayscale image dengan range dari 0 hingga 255. Range data seperti ini "terlalu besar" untuk model kita, apalagi dengan learning rate yang cukup kecil, sehingga kita perlu melakukan scaling dengan membaginya dengan 255. Sehingga kita dapatkan range data baru antara 0 dan 1.

1.6 Pengulasan tentang kode CNN

Halo kembali lagi di materi jaringan Saraf Convolutional canggih, dalam kesempatan ini kita akan melihat bagaimana sebuah kode dapat digunakan untuk menerapkan CNN dan karies dengan menggunakan metode data amneris. Apa yang akan dipelajari adalah betapa mudahnya kode tersebut dapat membaca dokumentasi Cairnes selama beberapa menit. Jadi strategi dalam membuat data jaringan saraf pada dasarnya hanya mendeklarasikan daftar surat yang ingin dipelajari dengan jaringan yang hanya satu baris dan kemudian plot biaya dan metrik lainya, namun ini semua terlihat mudah karena hanya sebuah penjelasan. Sebelum memulai pembelajaraan ini, kita dapat melihat fashion yang tinggi di repo, seperti yang kita lihat diatas ada beberapa impor

seperti jenis model yang kita inginkan secara berurutan dan semua jenis model yang diperlukan. Sebelumnya kami telah mempelajari semua jenis yang ada dimasa lalu sehingga kami dapat menemukan ide dan menggunakan untuk jenis model yang digunakan sekarang. Kami memiliki indikator untuk menentukan target yang berupa daftar indeks sehingga menjadi satu kesatuan yang berfungsi untuk mengategorikan data tersebut. jika pada dasarnya anda bisa menyalin kode apapun dari pembelajaraan sebelumnya yang dimuat dalam data yang sama yang akan dilakukan Karin. Untuk melakukan validasi split kereta. Tetapi disini saya mengacak-acak data untuk berjaga-jaga. Ingatlah bahwa kami ingin data berada dalam bentuk dan ketinggian dengan warna. Jadi kami membentuk ulang menjadi minus satu pada 28 satu. Karena gambar berukuran 28 x 28 dan skala abu-abu, kami juga ingin piksel gambar kami dinormalisasi sehingga kami membagi semuanya dengan 255 dan seperti halnya amnesti asli untuk mengatur label pada kolom pertama. Jadi X adalah segalanya mulai dari kolom 1 dan seterusnya. Dan mengapa semuanya ada di kolom 0. Selanjutnya kita mendapatkan K yang merupakan jumlah kelas dan pada baris berikutnya kita mengubah y menjadi matriks indikator sehingga kita dapat menggunakannya dalam cara yang sama dibagian kode selanjutnya.

1.7 VGG-16



Gambar 1.5 gambar VGG-16

Gambar diatas adalah arsitektur dari VGG16 (16 Layer). Yang akan kita gunakan adalah Feature Extraction Layer saja, tentu saja dengan weights yang dapat kita download. Weights nya berupa file .h5 seperti yang sudah kita gunakan sebelumnya.

FC layer pada VGG16 terdiri dari 4096–4096–4096 neuron pada hidden layer dan 1000 neuron pada output layer karena ImageNet mempunyai 1000 classs. Sedangkan dataset kita hanya mempunyai 2 class (Male/Female), sehingga kita harus membuat FC layer versi kita sendiri.

Kita akan menggunakan FC Layer yaitu 32 neuron pada hidden layer dan 1 neuron pada output dengan sigmoid activation (pada gambar 1.8).

VGG-16 Dependiencies and variable

namun pada hal ini kita membutuhkan package applications untuk dapat menggunakan VGG16.

gambar dibawah adalah salah satu contoh source code pada package applications dari VGG-16 Dependiencies and variable (pada gambar 1.9).

```
import numpy as np
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
from keras.optimizers import Adam
from keras.callbacks import TensorBoard

# Images Dimensions
img_width, img_height = 128, 128

train_data_dir = 'data/train'
validation_data_dir = 'data/validation'
nb_train_samples = 800
nb_validation_samples = 240
epochs = 50
batch_size = 16

# TensorBoard Callbacks
callbacks = TensorBoard(log_dir='./Graph')
```

Gambar 1.6 gambar VGG-16 Dependiencies and variable

VGG-16 Model and Data Augmentation

gambar dibawah adalah salah satu contoh source code pada package applications dari VGG-16 Model and Data Augmentation (pada gambar 1.10).

```
# Build VGG16
model = applications.VGG16(include_top=False, weights='imagenet')

# Training Data Augmentation
train_datagen = ImageDataGenerator(
    rescale=1. / 255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

# Rescale Testing Data
test_datagen = ImageDataGenerator(rescale=1. / 255)

# Train Data Generator
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')
train_features = model.predict_generator(
    train_generator, nb_train_samples // batch_size, verbose=1)
np.save('train_features.npy', train_features)

# Testing Data Generator
validation_generator = test_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
```

Gambar 1.7 gambar VGG-16 Model and Data Augmentation

Dengan menggunakan package applications kita bisa langsung menggunakan VGG-16 tanpa harus menyusunnya layer demi layer dan mendownload weights nya.

Argument include top sama dengan False diatas menandakan jika kita tidak menggunakan FC Layer dari VGG-16. Sehingga jika kita melakukan "predict" untuk model ini maka yang akan terjadi adalah dataset akan mengalir pada feature extraction layer dari VGG-16.

Hasilnya adalah feature map yang bisa kita simpan pada file train features.npy dan val features.npy yang nantinya bisa kita flatten dan kita gunakan untuk melakukan training pada FC Layer versi kita sendiri.

1.8 Convolutional Neural Network

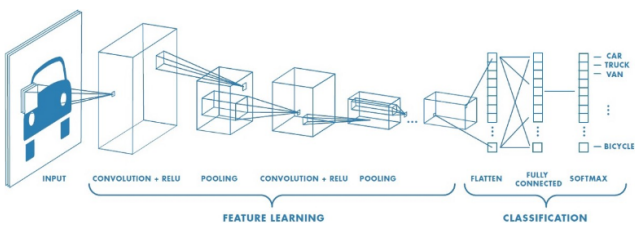
Convolutional Neural Network (CNN) adalah salah satu jenis neural network yang biasa digunakan pada data image. CNN bisa digunakan untuk mendeteksi dan mengenali object pada sebuah image.

Secara garis besar CNN tidak jauh beda dengan neural network biasanya. CNN terdiri dari neuron yang memiliki weight, biasa dan activation function seperti yang sudah kita pelajari pada part sebelumnya. Lalu apa yang membedakan? Arsitektur dari CNN dibagi menjadi 2 bagian besar, Feature Extraction Layer dan Fully-Connected Layer (MLP)

1.9 Feature Extraction Layer

Saya gunakan istilah ini karena proses yang terjadi pada bagian ini adalah melakukan "encoding" dari sebuah image menjadi features yang berupa angka-angka yang merepresentasikan image tersebut (Feature Extraction).

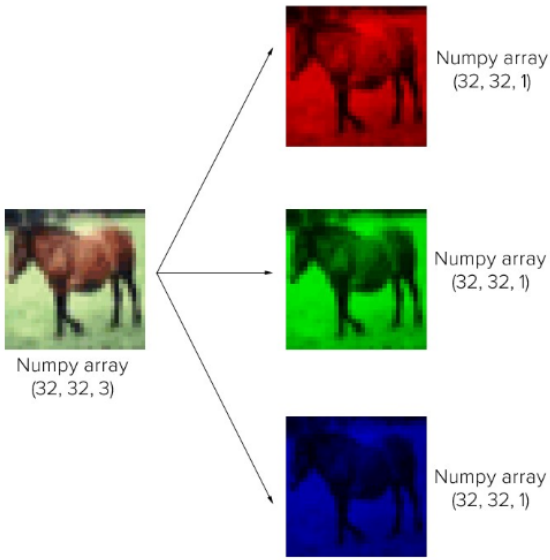
Feature extraction layer terdiri dari dua bagian. Convolutional Layer dan Pooling Layer. Namun kadang ada beberapa riset/paper yang tidak menggunakan pooling(pada gambar 1.8).



Gambar 1.8 Contoh FEL

1.10 Convolutional Layer

Gambar 1.9 adalah RGB (Red, Green, Blue) image berukuran 32x32 pixels yang sebenarnya adalah multidimensional array dengan ukuran 32x32x3 (3 adalah jumlah channel). Convolutional layer terdiri dari neuron yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (pixels). Sebagai contoh, layer pertama pada feature extraction layer biasanya adalah conv. layer dengan ukuran 5x5x3. Panjang 5 pixels, tinggi 5 pixels dan tebal/jumlah 3 buah sesuai dengan channel dari image tersebut. Ketiga filter ini akan digeser keseluruh bagian dari gambar. Setiap pergeseran akan dilakukan operasi "dot" antara input dan nilai dari filter tersebut sehingga menghasilkan sebuah output atau biasa disebut sebagai activation map atau feature map



Gambar 1.9 Contoh CL

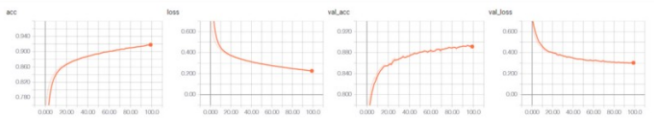
1.11 Training with TensorBoard Visualization

Kita akan menggunakan TensorBoard untuk melakukan visualisasi pada saat training. Seluruh training loss/accuracy dan validation loss/accuracy akan disimpan dan kita bisa melihat grafiknya.

Untuk menggunakan TensorBoard kita bisa gunakan command sebagai berikut :

```
tensorboard? ?logdir=Graph
```

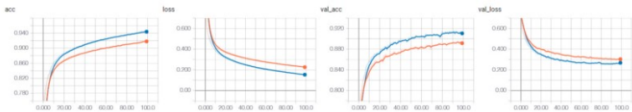
Listing 1.1 Contoh TBV



Gambar 1.10 Contoh TBV

1.12 Comparing Two Model

Dengan menggunakan TensorBoard, kita juga bisa membandingkan performa kedua model yang telah kita train.



Gambar 1.11 Contoh CTM

Grafik warna biru diatas adalah grafik dari model kedua yang menggunakan conv. layer yang lebih banyak. Bisa dilihat disitu kalau performa dari model ini jelas lebih bagus daripada model pertama.

Training dan Validation Loss yang didapatkan adalah 0.1521 dan 0.2571, sedangkan Training dan Validation Accuracy sebesar 94.46 persen dan 91.28 persen.

Sebenarnya kedua model ini masih bisa dioptimasi lagi, bisa dicoba pake learning rate yang lebih tinggi misalnya 0.001, tapi epoch lebih kecil lagi atau setting jumlah neuron pada FC Layer dan masih banyak lagi untuk improve performa dari model kita.

1.13 Transfer Learning

Transfer learning adalah suatu teknik atau metode yang memanfaatkan model yang sudah dilatih terhadap suatu dataset untuk menyelesaikan permasalahan lain yang serupa dengan cara menggunakannya sebagai starting point, memodifikasi dan mengupdate parameternya sehingga sesuai dengan dataset yang baru.

Kita akan gunakan model VGG-16 yang sudah dilatih pada data ImageNet dengan cara membuat arsitektur yang identik dengan VGG-16 tetapi tanpa fully-connected layer dan mendownload weights nya. Dengan menyediakan beberapa model ImageNet yang populer untuk bisa kita gunakan.

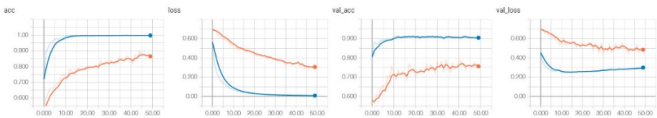
Seluruh weight VGG-16 telah dilatih menggunakan dataset ImageNet dan sudah dapat mengenali warna, tekstur, dll. Sehingga kita bisa manfaatkan ini untuk mengextract feature dari semua foto pada dataset kita.

1.13.1 Transfer Learning Result

```
import numpy as np
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
from keras.optimizers import Adam
from keras.callbacks import TensorBoard

# Images Dimensions
img_width, img_height = 128, 128

train_data_dir = 'data/train'
validation_data_dir = 'data/validation'
nb_train_samples = 800
nb_validation_samples = 240
epochs = 50
batch_size = 16

# TensorBoard Callbacks
callbacks = TensorBoard(log_dir='./Graph')

# Build VGG16
model = applications.VGG16(include_top=False, weights='imagenet')

```

Gambar 1.12 Transfer learning

Setelah 50 epoch training-testing, kita mendapatkan loss dan accuracy sebesar 0.2985 crossentropy loss dan 90.44 persen accuracy (pada gambar 1.12) .



Gambar 1.13 Grafik Learning

Namun jika dilihat dari grafiknya, masih terjadi overfitting. Kita bisa tangani ini dengan cara menggunakan FC Layer yang lebih sederhana, menggunakan Dropout, melakukan augmentasi yang lebih agresif lagi atau menambah jumlah data (pada gambar 1.13) .

1.13.2 ImageNet

ImageNet adalah sebuah dataset yang terdiri dari 1.200.000 gambar untuk training dan 100.000 untuk testing. Dataset ini terdiri dari 1000 classes jadi untuk setiap class ada 1.200 gambar.

Sebenarnya ImageNet challenge ini sudah dimulai sejak 2010, saya kurang paham algoritma dan model apa yang digunakan pada rentang 2010–2011. Namun hasil yang paling standout adalah AlexNet pada 2012. AlexNet adalah model pertama yang menggunakan Convolutional Neural Network (CNN).

Seiring dengan perkembangan teknologi tiap tahun, jumlah layer yang digunakan juga mengalami kenaikan yang hasilnya bisa dibilang setara dengan tingkat akurasi yang dihasilkan.

1.13.3 Data Augmentation

Gambar 1.14 Data Augmentation

Seperti yang sudah kita ketahui, untuk mendapatkan performa yang optimal, Deep Learning membutuhkan data yang lebih banyak dibandingkan dengan algoritma ML yang lain.

Dari dataset yang telah kita kumpulkan hanya terdapat 400 foto pria dan 400 foto wanita. Jumlah data tersebut masih kurang mencukupi untuk mendapatkan performa yang optimal.

Untuk itu kita perlu meng-augmentasi data tersebut. Data Augmentation adalah sebuah teknik memanipulasi sebuah data tanpa kehilangan inti atau esensi dari data tersebut. Untuk data berupa Image, kita bisa lakukan rotate, flip, crop, dll (Pada gambar 1.14).

```
# Training Data Augmentation
train_datagen = ImageDataGenerator(
    rescale=1. / 255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

# Rescale Testing Data
test_datagen = ImageDataGenerator(rescale=1. / 255)

# Train Data Generator
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')

# Testing Data Generator
validation_generator = test_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')
```

Gambar 1.15 Source Code Augmentation

Pada percobaan kita kali ini, kita akan melakukan shear, zoom dan flip sedangkan parameter rescale yang kita gunakan adalah membagi nilai RGB dari 0–255 dengan 255, sehingga kita mendapatkan nilai RGB pada rentang 0–1. Untuk data testing kita hanya melakukan rescale saja.

Method flow from directory dari ImageDataGenerator kita gunakan untuk mengubah data yang berupa "raw image" menjadi sebuah dataset yang akan kita gunakan untuk training dan testing, tentu saja dataset yang telah kita augmentasi tadi (Pada gambar 1.15).

1.14 Deeper Network

Kita bisa gunakan arsitektur yang lebih *deep* dengan menambahkan conv. layer. Tapi yang patut diperhatikan adalah semakin deep arsitektur yang kita gunakan semakin lama proses training karena semakin banyak parameter yang harus diupdate. Arsitektur yang seperti ini juga rawan terjadi overfitting.

Model kedua yang akan kita coba menggunakan ukuran filter yang sama yaitu 5x5 dan 3x3, namun kita gunakan stride yang lebih kecil dan kita melakukan dua kali downsampling.

1.15 Padding

Padding atau Zero Padding adalah parameter yang menentukan jumlah pixels (berisi nilai 0) yang akan ditambahkan di setiap sisi dari input. Hal ini digunakan dengan tujuan untuk memanipulasi dimensi output dari conv. layer (Feature Map).

Tujuan dari penggunaan padding adalah :

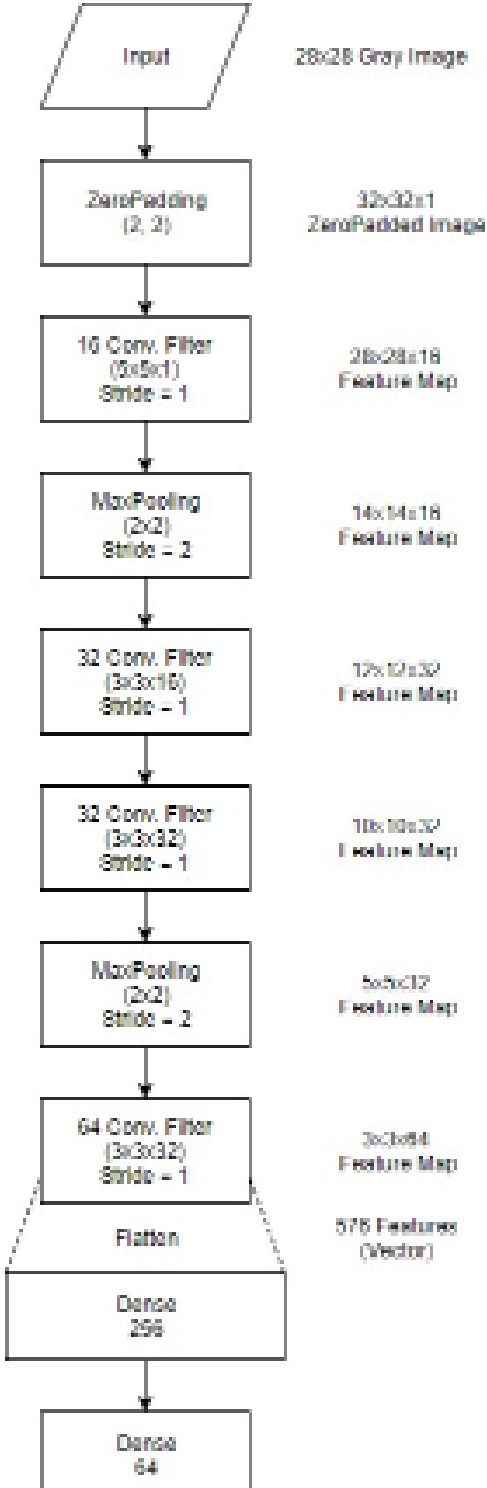
1. Dimensi output dari convolutional layer selalu lebih kecil dari inputnya. Output ini akan digunakan kembali sebagai input dari convolutional layer selanjutnya, sehingga makin banyak informasi yang terbuang. Dengan menggunakan padding, kita dapat mengatur dimensi output agar tetap sama seperti dimensi input atau setidaknya tidak berkurang secara drastis.

2. Meningkatkan performa dari model karena convolutional filter akan fokus pada informasi yang sebenarnya yaitu yang berada diantara zero padding tersebut, Sehingga kita bisa menggunakan convolutional layer yang lebih dalam sehingga lebih banyak features yang berhasil di extract.

1.16 Stride

Stride adalah parameter yang menentukan berapa jumlah pergeseran filter. Jika nilai stride adalah 1, maka conv. filter akan bergeser sebanyak 1 pixels secara horizontal lalu vertical.

Semakin kecil stride maka akan semakin detail informasi yang kita dapatkan dari sebuah input, namun membutuhkan komputasi yang lebih jika dibandingkan dengan


Input
28x28 Gray Image
ZeroPadding
(2, 2)
32x32x1
ZeroPadded Image
16 Conv. Filter
(5x5x1)
Stride = 1
28x28x16
Feature Map
MaxPooling
(2x2)
Stride = 2
14x14x16
Feature Map
32 Conv. Filter
(3x3x16)
Stride = 1
12x12x32
Feature Map
32 Conv. Filter
(3x3x32)
Stride = 1
10x10x32
Feature Map
MaxPooling
(2x2)
Stride = 2
5x5x32
Feature Map
64 Conv. Filter
(3x3x32)
Stride = 1
3x3x64
Feature Map
Flatten
576 Features
(Vector)
Dense
256
Dense
64

stride yang besar. Namun perlu diperhatikan bahwa dengan menggunakan stride yang kecil kita tidak selalu akan mendapatkan performa yang bagus.

1.17 KorNet

Arsitektur sederhana kita ini terdiri dari 87.969 buah parameter yang akan diupdate pada saat training. Pada feature extraction layer terdapat 4 Convolution Layer, ZeroPadding Layer dan MaxPooling Layer.

Pada fully-connected layer terdapat 2 buah layer dengan jumlah neuron masing-masing sebanyak 32 dan 1. Perlu diingat bahwa layer terakhir adalah output layer. 1 buah layer disini karena kita akan melakukan binary classification, 0 untuk pria dan 1 untuk wanita. Sehingga activation function yang harus kita gunakan pada output layer adalah sigmoid dengan loss function binary crossentropy. Kita juga bisa menggunakan 2 neuron pada output layer, menggunakan activation function softmax dan loss function categorical crossentropy seperti pada Part-7.

1.18 ResNet

Model ResNet merupakan model yang menggunakan deep residual learning framework. Dengan menggunakan framework ini, setiap layer network memiliki referensi ke layer network sebelumnya; hal ini menjadikan proses optimasi menjadi lebih mudah daripada layer network-layer network yang tidak memiliki keterhubungan. Karena proses optimasi yang lebih mudah, neural network yang dibentuk dapat memiliki jumlah layer yang banyak sampai dengan 34 layer dan akibatnya, akurasi meningkat dari neural network yang tidak menggunakan residual network.

1.19 Pooling Layer

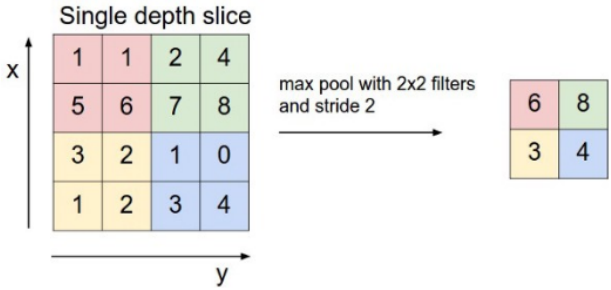
Pooling layer biasanya berada setelah convolutional layer. Pada prinsipnya pooling layer terdiri dari sebuah filter dengan ukuran dan stride tertentu yang akan bergeser pada seluruh area feature map.

Pooling yang biasa digunakan adalah Max Pooling dan Average Pooling. Sebagai contoh jika kita menggunakan Max Pooling 2x2 dengan stride 2, maka pada setiap pergeseran filter, nilai maximum pada area 2x2 pixel tersebut yang akan dipilih, sedangkan Average Pooling akan memilih nilai rata-ratanya.

Keuntungan dari ResNets adalah:

1. kinerja tidak menurun dengan jaringan yang sangat dalam
2. lebih murah untuk dihitung
3. kemampuan untuk melatih jaringan yang sangat dalam

ResNet berfungsi karena:



Gambar 1.17 Contoh PL

1. mengidentifikasi fungsi mudah untuk blok residual untuk dipelajari

2. menggunakan koneksi-loncatan membantu gradien untuk kembali-menyebar dan dengan demikian membantu Anda untuk melatih jaringan yang lebih dalam

1.20 Fully Connected Layer

Feature map yang dihasilkan dari feature extraction layer masih berbentuk multi-dimensional array, sehingga kita harus melakukan flatten atau reshape feature map menjadi sebuah vector agar bisa kita gunakan sebagai input dari fully connected layer.

Fully Connected Layer yang dimaksud disini adalah MLP yang sudah pernah kita pelajari sama-sama pada bagian ke-4 dan bagian ke-5. Fully Connected Layer memiliki beberapa hidden layer, activation function, output layer dan loss function.

1.21 The Problem

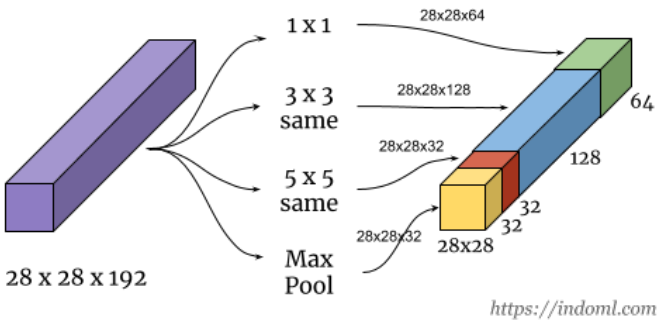
Data MNIST diatas terdiri dari nilai 0–255 untuk setiap pixel yang ada. Kita bisa saja menggunakan MLP untuk melakukan klasifikasi untuk semua digit dengan hasil yang cukup baik karena sebagian besar data pada MNIST, object yang akan dikenali berada ditengah-tengah gambar.

Lalu bagaima jika object yang akan dikenali tidak berada ditengah-tengah gambar? Disinilah kelemahan dari MLP. Angka 6 yang berada ditengah-tengah gambar akan berhasil dikenali, tetapi angka 6 yang berada dipojok kiri mungkin tidak akan dikenali.

Kita bisa menggunakan data yang sangat banyak dengan tiap digit berada pada lokasi yang berbeda, namun ini bukan cara yang efisien untuk mengatasi permasalahan tersebut.

1.22 Inception

Motivasi dari jaringan awal adalah, daripada mengharuskan kita untuk memilih ukuran filter secara manual, biarkan jaringan memutuskan apa yang terbaik untuk dimasukkan ke dalam sebuah layer. Kami memberikan pilihan dan semoga akan mengambil yang terbaik untuk digunakan di lapisan itu (pada gambar 1.18) .



Gambar 1.18 gambar Inception

BAB 2

GARIS BESAR DAN PERSPEKTIF

BAB 3

GARIS BESAR DAN PERSPEKTIF

Selamat datang kembali ke kelas ini jaringan saraf convolutional canggih. Jadi mari kita bicara tentang kursus ini dan mengapa ini berbeda dari jaringan saraf convolutional pertama yang merupakan pembelajaran mendalam di Python bagian 3 dalam kursus sebelumnya Anda diperkenalkan pada konvolusi. Kami memiliki seluruh bagian yang ditujukan untuk hanya belokan dengan sendirinya melihat cara kerjanya. Melihat beberapa cara untuk menggunakannya untuk beberapa aplikasi klasik begitu kita memiliki landasan yang baik pada konvolusi kami menambahkannya ke jaringan saraf yang tentu saja memberi kita jaringan saraf convolutional.

Kami melihat bahwa ini dapat digunakan untuk membangun pengklasifikasi gambar yang jauh lebih kuat daripada feedforward biasa jaring karena filter yang ditemukan dengan merambat kembali melalui konvolusi memungkinkan kita mencari fitur gambar seperti tepi yang berbeda sudut dan warna serta tekstur yang berbeda. Singkatnya menggunakan konvolusi memungkinkan kita untuk mengambil keuntungan dari struktur gambar sedangkan feedforward net lebih umum. Itu tidak mengasumsikan apa-apa tentang data tetapi itu membuatnya kurang cocok untuk gambar pada khususnya.

Dan mereka perlu melakukannya dengan aplikasi cepat. Salah satu yang paling populer adalah transfer gaya. Di sinilah Anda mengambil satu gambar yang kami

sebut konten. Gambar lain yang kami sebut gaya dan kami menggabungkannya. Gambar akhir memiliki konten dari satu gambar tetapi gaya yang lain. Seolah-olah Anda menyewa seorang pelukis untuk melukis cakrawala tetapi melakukannya dengan menggunakan gaya malam berbintang yang Anda bisa bayangkan bahwa ini akan memakan waktu seorang pelukis manusia sejati. Sedangkan jaringan saraf dapat melakukannya dalam hitungan detik dan itu terlihat hebat. Jadi tema utama dari kursus ini adalah bahwa dalam kursus sebelumnya kami melihat apa yang terjadi di dalam saraf jaringan khusus. Sekarang kita berada di luar jaringan saraf. Kami ingin bertanya apa yang bisa kami lakukan dengan CNN. Sistem apa yang dapat berisi jaringan saraf dan bagaimana kita dapat menggunakan pengetahuan kita tentang jaringan saraf untuk membuat mereka lebih baik. Terima kasih telah mendengarkan dan sampai jumpa di kuliah berikutnya.

BAB 4

TINJAUAN

4.1 Tinjauan

Konvolusi

1. Filter (3x3 dalam gambar) adalah tensor berat yang dipelajari dengan backpropagation
2. Kesalahan umum: pikir kami merancang filter untuk menjadi detektor tepi, dll
3. Bagian yang tumpang tindih akan dikalikan elemen-bijaksana dijumlahkan
4. Secara konsep setara dengan ”produk matrix dot”
5. Jika filter berkorelasi dengan gambar, output akan menjadi jumlah yang sangat besar
6. Jika berbeda, nilai output akan menjadi kecil
7. Jadi, cara yang sama validnya untuk memikirkannya adalah ”cross-corelation”