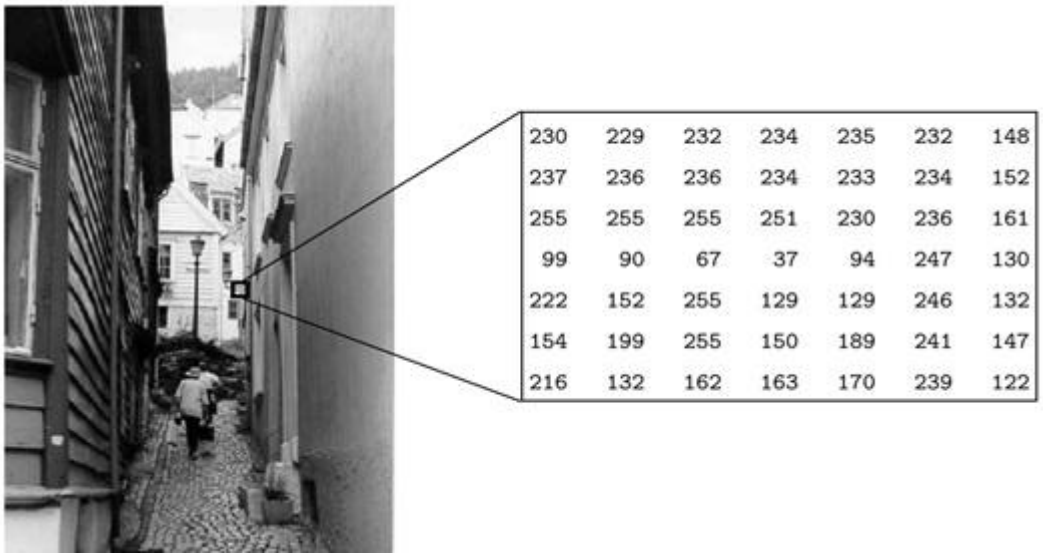
Algoritma CNN

Oleh : Oscar Hadikaryana, ST, MT

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. CNN adalah sebuah teknik yang terinspirasi dari cara mamalia — manusia, menghasilkan persepsi visual.

Secara garis besar Convolutional Neural Network (CNN) tidak jauh beda dengan neural network biasanya. CNN terdiri dari neuron yang memiliki weight, bias dan activation function. Convolutional layer juga terdiri dari neuron yang tersusun sedemikian rupa sehingga membentuk sebuah filter dengan panjang dan tinggi (pixels).

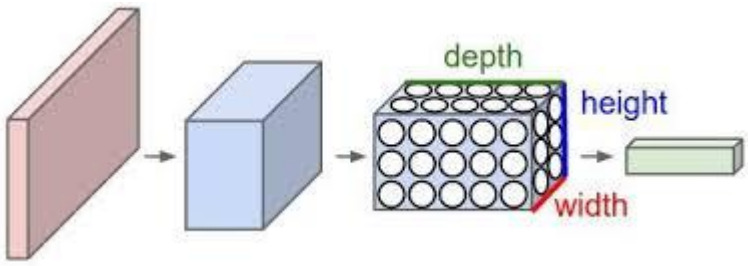
Algoritma CNN adlah MLP (Multi Layer Perceptron) yang didisain secara khusus untuk mendidentifikasi image/citra dua dimensi. CNN meniru cara kerja otak manusia untuk mengenali objek yang dilihatnya. Dengan bantuan CNN, komputer dapat melihat dan membedakan berbagai objek atau dengan kata lain komputer dapat mepresepsikan sebuah citra.



Gambar Citra dan nilai pixel untuk citra digital grayscale
https://catatanpeneliti.wordpress.com/2013/06/04/empat-tipe-dasar-citra-digital/

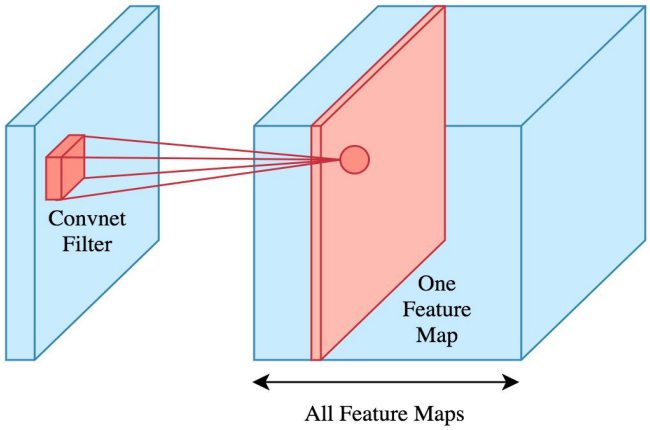
Bentuk array sangat bergantung pada resolusi dan ukuran gambar, dicontohkan citra dalam format jpg mempunyai resolusi 480 x 480 maka dalam bentuk array menjadi 480 x 480 x 3. Angka 3 ini merupakan nilai RGB (Red Green Blue).

Jika array digambar dalam bentuk model persegi maka terbentuk sebuah volume. Misalkan citra dengan ukuran 240 x 360 x 3 akan membentuk volume kubus, 240 (panjang) x 360 (lebar) x 3 (tinggi/kedalaman). Kita lihat dalam ilustrasi :



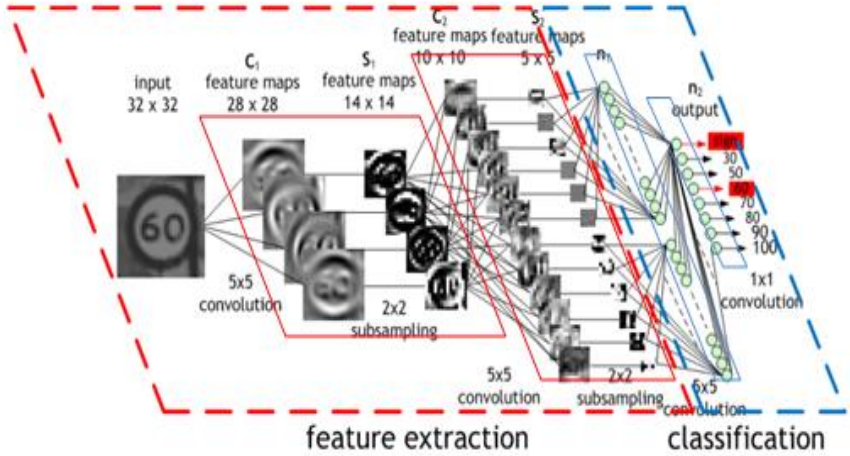
Gambar Arsitektur CNN secara umum
(http://it.maranatha.edu/wp-content/uploads/2016/01/muftah-cnn-anjing.pdf)

Secara garis besarnya, CNN memanfaatkan proses konvolusi dengan menggerakan sebuah kernel konvolusi (filter) berukuran tertentu ke sebuah gambar, komputer mendapatkan informasi representatif baru dari hasil perkalian bagian gambar tersebut dengan filter yang digunakan.



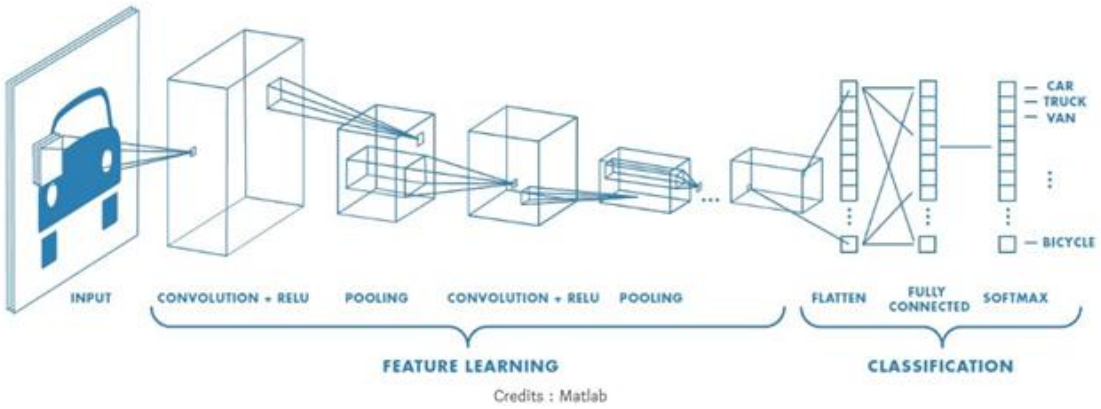
Gambar Convolution net Layer dengan satu l buah filter
https://www.kdnuggets.com/2019/07/convolutional-neural-networks-python-tutorial-tensorflow-keras.html

CL umumnya lebih dari satu filter, misal ada empat filter, maka CL akan berisi sejumlah neurons yang tersusun dalm kisi berukuran 28 x 28 x 4.



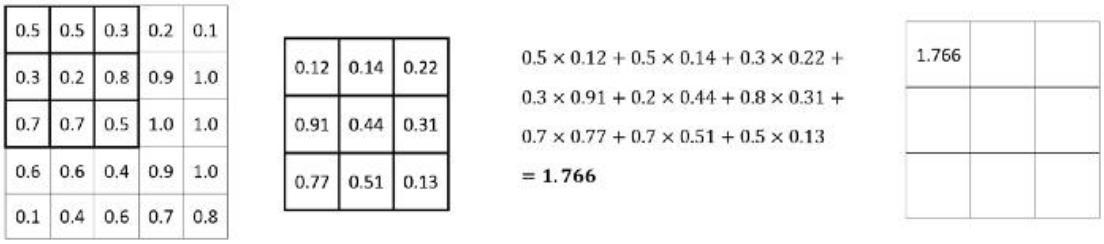
Gambar Contoh Arsitektur CNN
(https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b)

Jika di analogikan dengan fitur dari wajah manusia, layer pertama merupakan refleksi goresan-goresan berbeda arah, pada layer kedua fitur seperti bentuk mata, hidung, dan mulut mulai terlihat, hal ini karena di lakukan pooling/penggabungan dari layer pertama yang masih berupa goresan-goresan, pada layer ketiga akan terbentuk kombinasi fitur-fitur mata, hidung, dan mulut yang nantinya akan disimpulkan dengan wajah orang tertentu.
Pada gambara diatas, dengan input berupa citra yang dijadikan vektor tunggal 32 × 32. Di tiap hidden layer, terdapat beberapa neuron layaknya empat feature maps C1 pada gambar tersebut. Neuron-neuron pada C1 dihubungkan dengan neuron di S1, dan seterusnya. Lapisan terakhir yang terhubung dengan lapisan-lapisan tersembunyi sebelumnya disebut dengan output layer dan merepresentasikan hasil akhir klasifikasi kelas.
Keseluruhan skala dalam objek sangat penting agar input tidak kehilangan informasi spasialnya yang akan diekstraksi fitur dan diklasifikasikan. Hal ini akan menambah tingkat akurasi dan optimum algoritma CNN. Seperti pada kubus yang memiliki skala pada panjang, lebar, dan tinggi.
CNN terdiri dari berbagai lapisan yang dimana setiap lapisan memiliki Application Program Interface (API) alias antarmuka program aplikasi sederhana. Pada Gambar 4, CNN dengan input awal balok tiga dimensi akan ditransformasikan menjadi output tiga dimensi dengan beberapa fungsi diferensiasi yang memiliki atau tidak memiliki parameter. CNN membentuk neuron-neuronnya ke dalam tiga dimensi (panjang, lebar, dan tinggi) dalam sebuah lapisan.



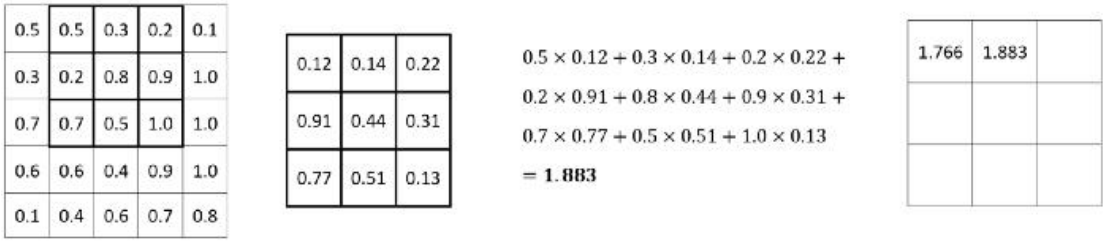
https://medium.com/@nadhifasofia/1-convolutional-neural-network-convolutional-neural-network-merupakan-salah-satu-metode-machine-28189e17335b

Mari kita lihat operasi konvolusi yang sedang beraksi. Kami memiliki gambar 5x5 dan filter 3x3. Filter meluncur (melilit) di atas setiap kumpulan piksel 3x3 dalam gambar, dan menghitung perkalian elemen-bijaksana. Hasil perkalian kemudian dijumlahkan:

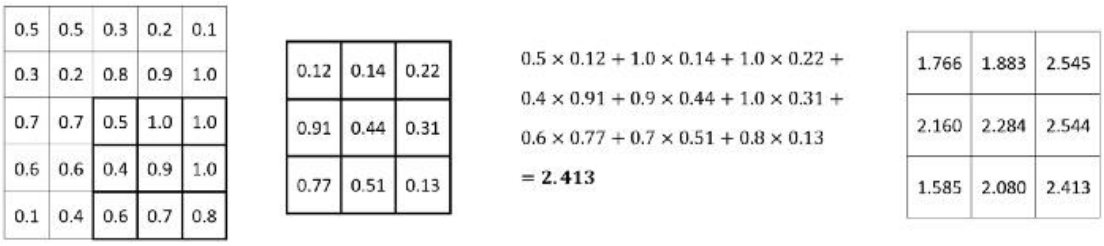


https://towardsdatascience.com/tensorflow-for-computer-vision-how-to-implement-convolutions-from-scratch-in-python-609158c24f82

Proses ini diulang untuk setiap set 3x3 piksel. Berikut perhitungan untuk set berikut:



Itu terus berlanjut sampai set akhir 3x3 piksel tercapai:



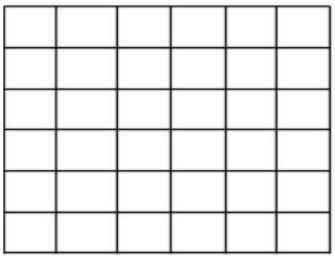
Padding dalam CNN

Padding secara umum berarti bahan bantalan. Di CNN ini mengacu pada jumlah piksel yang ditambahkan ke gambar saat sedang diproses yang memungkinkan analisis yang lebih akurat. Padding ini menambahkan beberapa ruang ekstra di sekitar gambar yang membantu kernel untuk meningkatkan kinerja. Ini lebih membantu bila digunakan untuk mendeteksi batas suatu gambar.

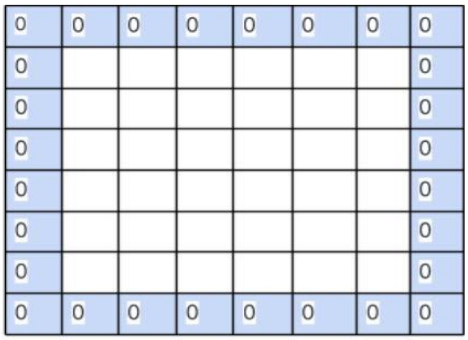
Ketika gambar sedang menjalani proses konvolusi kernel dilewatkan sesuai dengan langkahnya. Saat bergerak, kernel memindai setiap piksel dan dalam proses ini memindai beberapa piksel beberapa kali dan beberapa piksel lebih sedikit (perbatasan). Secara umum, piksel di tengah lebih sering digunakan daripada piksel di sudut dan tepi. Hal ini pada gilirannya dapat menyebabkan deteksi perbatasan yang buruk. Kita bisa mengatasi masalah ini dengan menggunakan padding.

Untuk citra gray scale (n x n) dan filter/kernel (f x f) maka dimensi citra yang dihasilkan dari operasi konvolusi adalah ((n — f) + 1) x ((n — f) + 1)). Sebagai contoh jika kita menggunakan citra 8x8 dan filter 3x3 outputnya akan menjadi 6x6 setelah konvolusi. Ini berarti setelah setiap konvolusi gambar menyusut. Hal ini dapat menyebabkan keterbatasan untuk membangun jaringan yang lebih dalam tetapi kita dapat mengatasinya dengan padding.

Ada beberapa jenis padding seperti Valid, Same, Causal, Constant, Reflection dan Replication. Dari jumlah tersebut yang paling populer adalah padding yang valid dan padding yang sama. Mari kita lihat mereka lebih jelas.



6x6 image

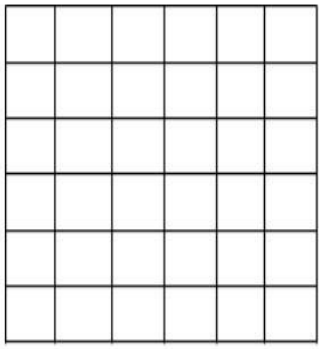


6x6 image with 1 layer of zero padding

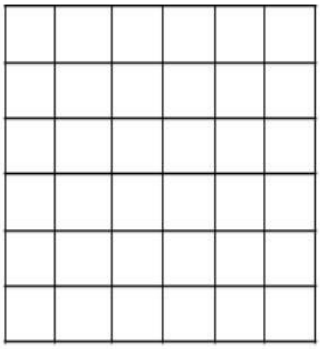
https://mahithas.medium.com/how-padding-helps-in-cnn-2b87957e1b

Valid Padding- Padding yang valid (atau tanpa padding): Padding yang valid adalah tanpa padding. Ini secara default keras pilih jika tidak ditentukan. Ketika (n x n) gambar digunakan dan (f x f) filter digunakan dengan padding yang valid, ukuran gambar keluaran akan menjadi (n-f+1)x(n-f+1).

[(n x n) gambar] * [(f x f) filter] -> [(n — f + 1) x (n — f + 1) gambar]



6x6 image



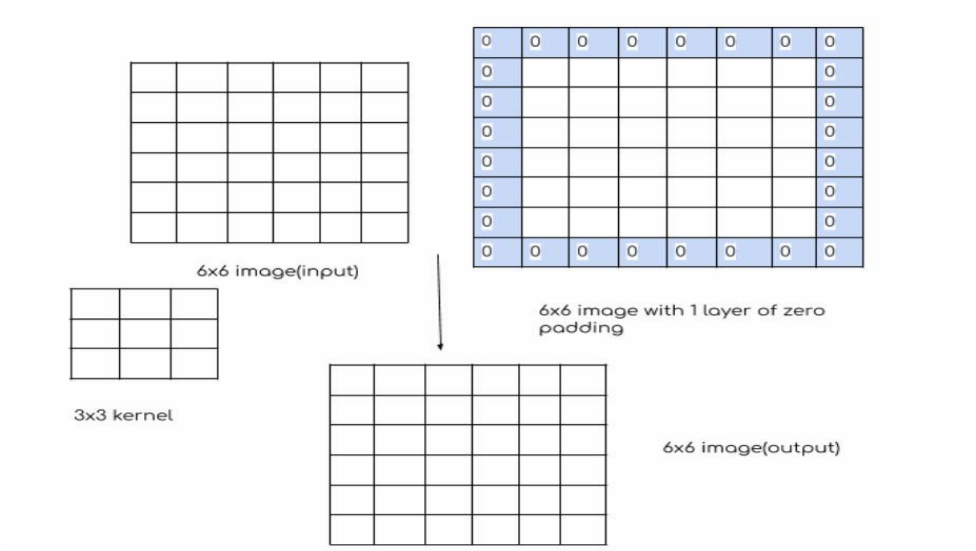
6x6 valid padding(no padding)

Same Padding- Padding yang sama: Padding yang sama digunakan ketika kita membutuhkan output dengan bentuk yang sama dengan input. Nilai ini menghitung dan menambahkan bantalan yang diperlukan ke gambar input untuk memastikan bentuk sebelum dan sesudahnya. Jika nilai untuk padding adalah nol maka dapat disebut sebagai padding nol. Ketika padding disetel ke nol, maka setiap piksel dalam padding memiliki nilai nol. Ketika padding nol diatur ke 1 maka 1 batas piksel ditambahkan ke gambar dengan nilai nol. Saat kita menggunakan gambar (n x n) dan filter (f x f) dan kita menambahkan padding (p) ke gambar. Ukuran gambar keluaran adalah (n x n). Itu berarti mengembalikan ukuran gambar. Persamaan berikut mewakili ukuran input dan output dengan padding yang sama.

[(n + 2p) x (n + 2p) gambar] * [(f x f) filter] -> [(n x n) gambar].

Nilai p = (f-1)/2 karena (n+2p-f+1) = n

Kita dapat menggunakan rumus di atas dan menghitung berapa banyak lapisan padding yang dapat ditambahkan untuk mendapatkan ukuran yang sama dari gambar aslinya. Misalnya jika kita menggunakan image 6x6 dan filter 3x3 kita membutuhkan 1 layer padding [P = (3 -1)/2 = 1] untuk mendapatkan output image 6x6. Contoh ini direpresentasikan dalam diagram berikut.



Penting untuk memahami konsep padding karena membantu kita menjaga informasi batas dari data input. Karena batas aslinya tidak dapat diperiksa dengan benar karena batas tidak dapat berada di tengah kernel untuk dipindai dengan baik. Oleh karena itu perlu adanya padding agar lebih akurat. Ini juga membantu untuk mempertahankan ukuran input. Parameter untuk padding bisa valid atau sama.

Ref : https://mahithas.medium.com/how-padding-helps-in-cnn-2b87957e1b

Ukuran citra masukan dapat dinyatakan sebagai :

$w_1 \times h_1 \times d_1$

Dimana w adalah lebar, h adalah tinggi dan d adalah jumlah kana Red Green Blue (RGB) atau kedalaman citra masukan.
CNN memiliki 4 hyperparameters, yitu : jumlah filter K, ukuran bidang reseptif atau spatial extent (tinggkat spasial) F, lebar langkah atau stride s, dan zero padding P.

Volume keluar (output) dapat dihitung menggunakan formula :

$w_2 \times h_2 \times d_2$

Dimana :

$$w_2 = \frac{(w_1 - F + 2P)}{S} + 1$$

$$h_2 = \frac{(h_1 - F + 2P)}{S} + 1$$

$$d_2 = K$$

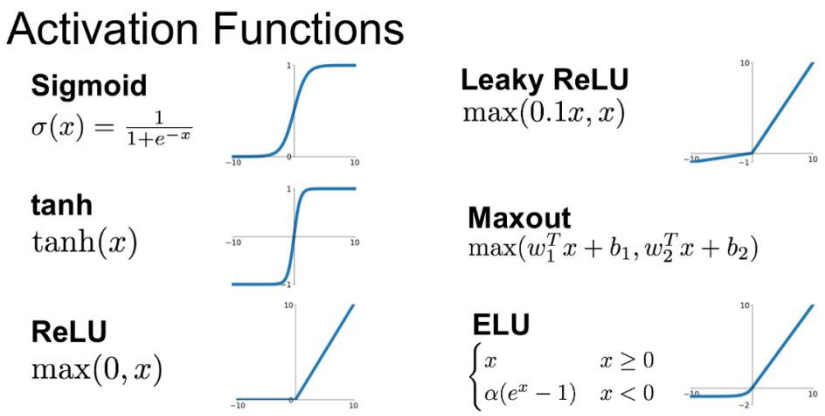
Biasanya CNN menggunakan seting hyperparameters :
1. Jumlah filter K berupa bilangan pangkat 2, misal 32, 64, 128, 512
2. Ukuran Filter :

F = 3	S = 1	P = 1	
F = 5	S = 1	P = 2	
F = 5	S = 2	P = berapun yang sesuai	
F = 1	S = 1	P = 0	(konvolusi 1 x 1)

Secara umum CNN tidak didisain untuk tidak menurunkan ukuran data terlalu cepat. Karena akan menurun performasi CNN.

Activation Layer/ ReLu Layer

Activation Layer terletak sebelum melakukan *Pooling Layer* dan setelah melakukan *Convolution Layer*. ReLu layer atau Rectified Linier Units ini mengaplikasikan fungsi aktifasi f(x) = max(0,x). Pada lapisan ini, terjadi proses pengubahan nilai nilai *feature map* pada jarak tertentu tergantung pada *Activation Function* yang dipakai. Tujuan utamanya yaitu untuk meneruskan nilai yang memperlihatkan *dominant feature* dari gambar inputan. Terdapat beberapa *Activation Function* yang kerap dipakai untuk *Convolutional Network*. Berikut adalah beberapa *activation function* tersebut.



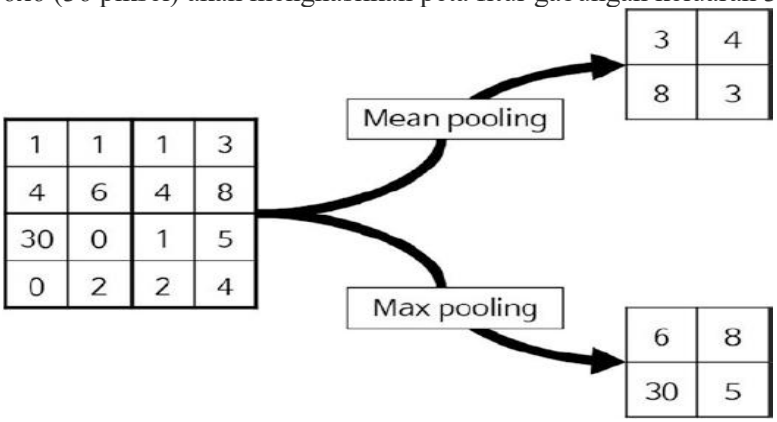
https://kotakode.com/blogs/2707/Convolutional-Neural-Network-(CNN)

Pooling Layer

Pooling layer adalah layer baru yang ditambahkan setelah convolutional layer. Secara khusus, setelah nonlinier (misalnya ReLU) telah diterapkan ke output peta fitur oleh lapisan konvolusi. Lapisan penyatuan beroperasi pada setiap peta fitur secara terpisah untuk membuat satu set baru dengan jumlah yang sama dari peta fitur yang dikumpulkan.

Pooling melibatkan pemilihan operasi pooling, seperti filter yang akan diterapkan pada peta fitur. Ukuran operasi penyatuan atau filter lebih kecil dari ukuran peta fitur; khusus, hampir selalu 2x2 piksel diterapkan dengan langkah 2 piksel.

Ini berarti bahwa lapisan penyatuan akan selalu mengurangi ukuran setiap peta fitur dengan faktor 2, mis. setiap dimensi dibelah dua, mengurangi jumlah piksel atau nilai di setiap peta fitur menjadi seperempat ukuran. Misalnya, lapisan penyatuan yang diterapkan ke peta fitur 6x6 (36 piksel) akan menghasilkan peta fitur gabungan keluaran 3x3 (9 piksel).



https://kotakode.com/blogs/2707/Convolutional-Neural-Network-(CNN)

Operasi penyatuan ditentukan, bukan dipelajari. Dua fungsi umum yang digunakan dalam operasi pooling adalah:

1. Average Pooling: Hitung nilai rata-rata untuk setiap patch pada peta fitur.
2. Max pooling- Penggabungan Maksimum : Hitung nilai maksimum untuk setiap tambalan peta fitur.

Hasil dari penggunaan lapisan penyatuan dan pembuatan peta fitur yang disampel atau dikumpulkan adalah versi ringkasan dari fitur yang terdeteksi di input. Mereka berguna karena perubahan kecil pada lokasi fitur dalam input yang terdeteksi oleh lapisan konvolusi akan menghasilkan peta fitur yang digabungkan dengan fitur di lokasi yang sama. Kemampuan yang ditambahkan dengan pooling ini disebut model invariance to local translation.

Normalization Layer

Normalization Layer berguna untuk mengatasi perbedaan rentang nilai yang siginifikan pada citra masukan. Banya sekali metode yang diusulkan oleh para ilmuwan, tetapi belakang tidak banyak digunakan dengan melihat dampaknya yang relatif kecil bahkan tidak ada dampak.

Fully Connected Layer (FCL)

Pada lapisan yang terhubung penuh (FCL), setiap neurons memiliki koneksi penuh ke semua aktivasi dalam lapisan sebelumnya. Modelnya sama dengan Multi Layer Perceptron (MLP) , aktivasinya sama dengan MLP, yaitu komputasi menggunakan suatu perkalian matriks yang diikuti dengan bias offset.
Jaringan saraf adalah seperangkat fungsi non-linier yang bergantung. Setiap fungsi individu terdiri dari neuron (atau perceptron). Pada lapisan yang terhubung penuh, neuron menerapkan

transformasi linier ke vektor input melalui matriks bobot. Transformasi non-linier kemudian diterapkan pada produk melalui fungsi aktivasi non-linier f.

$$y_{jk}(x) = f\left(\sum_{i=1}^{n_H} w_{jk}x_i + w_{j0}\right)$$

Sederhananya kita mengambil produk titik antara matriks bobot W dan vektor input x. Suku bias (W0) dapat ditambahkan di dalam fungsi non-linier.

Loss Layer

Ref ;

DR. Suyanto, ST, MSc., Deep Learning-Modernisasi Machine Learning untuk Big Data, Pnb. Informatika, Bandung, 2019
https://towardsdatascience.com/tensorflow-for-computer-vision-how-to-implement-convolutions-from-scratch-in-python-609158c24f82
https://mahithas.medium.com/how-padding-helps-in-cnn-2b87957e1b
https://www.tensorflow.org/tutorials/images/cnn
https://colab.research.google.com/
https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/