

Images

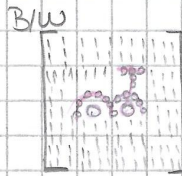
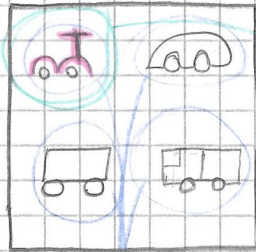
Image classification

→ Dataset for image classification some size pictures

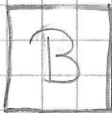
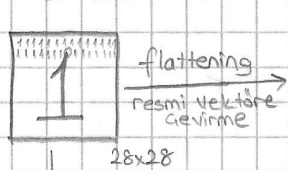
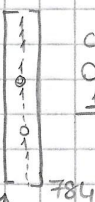
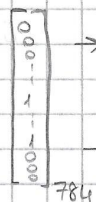
 $\begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$
mxn

28x28

→ elimdeki resimler mümkünse kare ve hepsi aynı boyutta olmalı.

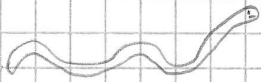
Image ProcessingRaw $\left\{ \begin{array}{l} \text{B/W} \\ \text{Grayscale} \\ \text{RGB} \end{array} \right.$ derivatives $\nabla = \text{edges}$
 $H = \text{corners}$ Object DetectionÖnce resimdeki nesneleri cebime koyayım
sonra object/image classification yapmam lazım.bunun matrise yansıma şekli
(Kenar) çizgi olan yerler = 0, beyaz yerler (arka plan) = 1

Buradaki objeleri al, aynı boyutta yap. Nasıl reshape yapıcam?

 $\begin{bmatrix} 100 & 150 & 150 & 100 \\ 200 & 250 & 250 & 200 \end{bmatrix}$
4x2boyutunu küçültmek istesem
4x2 → 2x2 yapıcamikililerin
ortalamasını
al, yaz. $\begin{bmatrix} 125 & 125 \\ 225 & 225 \end{bmatrix}$ $\begin{bmatrix} 100 & 150 & 150 & 100 \\ 200 & 250 & 250 & 200 \end{bmatrix}$
4x2büyütmek istesem
ortalamasını alıp
ortasına yaz $\begin{bmatrix} 100 & 125 & 150 & 150 & 125 & 100 \\ 200 & 225 & 250 & 250 & 225 & 200 \end{bmatrix}$ * Gözünürlük ne çok küçük ne çok büyük olacak. Çok küçük olursa hepsi birbirine benzer.
Detayları yakalayıp karar veremez. $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ çok küçük
olsalar $\begin{bmatrix} 8 & 8 & 8 \\ 8 & 8 & 8 \\ 8 & 8 & 8 \end{bmatrix}$ anlaşılmıyor→ bunun için 1024x1024'e gerek yok.
16x16 ya da 28x28 anlayabilmemiz için yeterli.
Çok yüksek gözünürlüğe gerek yok.Mnist Datasetyazı yazan yerler
0 alınıyorconvert
0 → 1
1 → 0→ bu vektörü analiz etmek diğerinden
daha kolaydır.

→ bunda resimde yazı yazan yerler 1

→ Kedi ile yılanı nasıl ayırt edersin?



→ Yılan

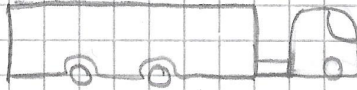
Think Simple



→ Kedi

→ Kulak, bacak ayırt edici

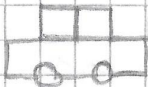
→ Farklı taşıtları nasıl ayırt edersin?



truck



sport car



sedan



bicycle

→ Camları ve tekerlekleri ayırt edici.

Glass

Tyre

[1]

[1 1 1]

[1]

[1 1]

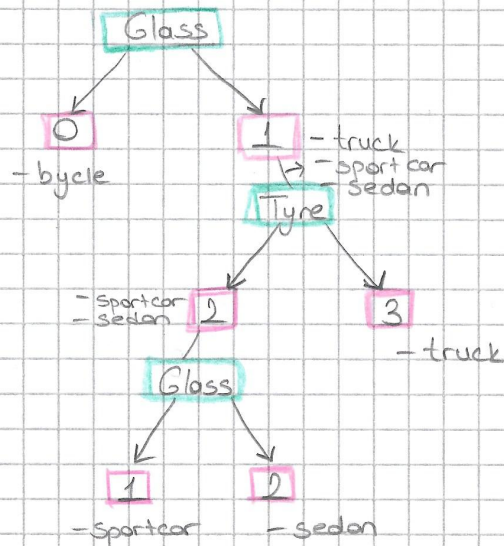
[1 1]

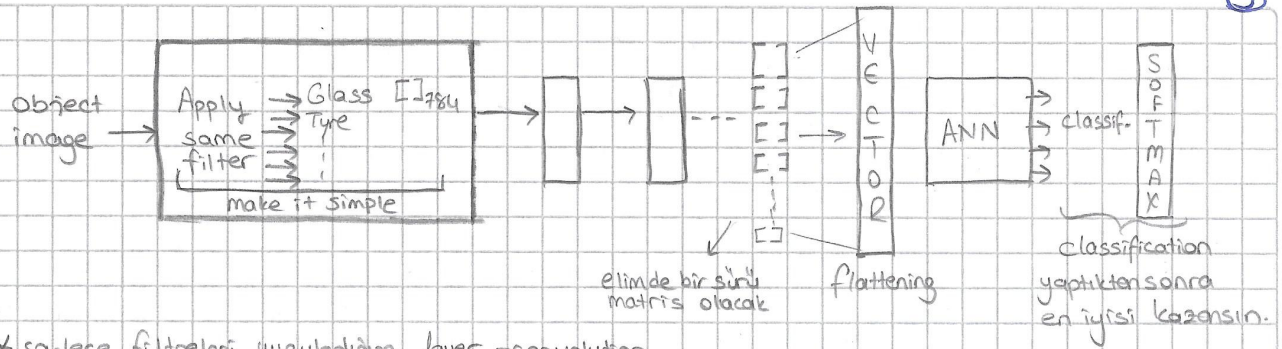
[1 1]

[0 0 0 0]

[1 1]

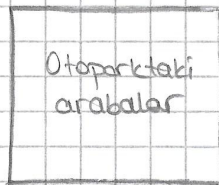
Basit DT ile gösterirsek;






* sadece filtreleri uyguladığım layer = convolution.

→ Resmi al, uygun filtrelerden geçir, sadeleştir (pooling), (filtre → pooling) bunu bir kaç kez yaparız, bunun sonucunda oluşan vektörü al, ANN'e koy, sınıflandır ve kazananı belirlemek için softmax'e koy.



→ Resimde kaç tane araba olduğunu bulmak → image processing

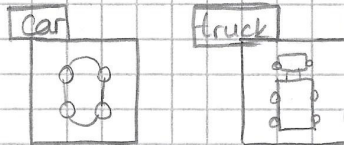
Türev al, kenarları bul,  kare içine al, say.

! Bunun için DL'e gerek yok !

OpenCV

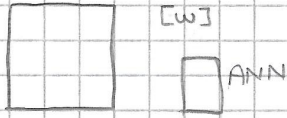
- Image processing
- object detection
- object classification

Kenarları bul, box içine al, detect et → image processing
* hangi sınıf olduğunu yaz. → bu DL ile ilgili



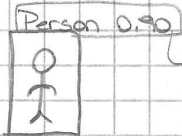
→ Aracın rengi asfaltla çok yakınsa ve eğer ışık çok az ya da yoksa renk asfalttan ayırt edilemez böyle olunca da türev çok küçük çıkar. Bu yüzden de onun araba olduğunu ya da orada herhangi bir şey olduğunu anlayamaz, kaçırır. ⇒ Hata

Yolo v5



Yolo - You only live once

Yolo train edildiği için ağırlıkları ile oluşturulmuş kernel filtreleri belirlir. Bir objecti detect edip yolo'ya ver, o sana ne olduğunu (hangi class'a ait olduğunu) söyler. İlla detect etmene de gerek yok. Yolo'ya bir video ver (onun içinde detect özelliği de var) kendi detect edip ne olduğunu söylüyor.



Person 0.90
1/90 ihtimalle
bu insan diyor

→ Git'te yolo var alıp kullanabilirsin ama her şey için kullanamassın.

Yolo → object classification for traffic

→ Yolo weightleri çıkmış CNN örneğidir.

→ Yolo VGG- ResNet gibi modelleri kullanıp kendi modelini kurmuş.

Keras * Conv2D *

model.add(Conv2D(32,3,3), activation="relu", padding="same", input_shape=(150,150,3))

Conv2D(32,3,3) : 32 tane 3x3 lük convolution filtre var.

Convolution, filtreleme yapan layer, filtreler random oluşuyor.

padding = 'kırk tane atlayarak yaptığı

= same olursa aynı, shift etmiyorsun demek.

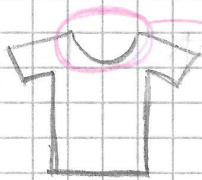
input_shape(150,150,3) : 150x150 resim, 3 renk → RGB

- Mnist data olsaydı (28,28,1) olurdu → 28x28, 1 renk

activation="relu" : - Relu 0'dan küçük olan değerleri sıfırlıyor. Büyük olanları sınıfa atıyor.

1	-1	-1
-1	1	-1
-1	-1	1

→ 0'dan küçük olanları 0 yapar.



bunu şu kernel filteri ile detect eder.

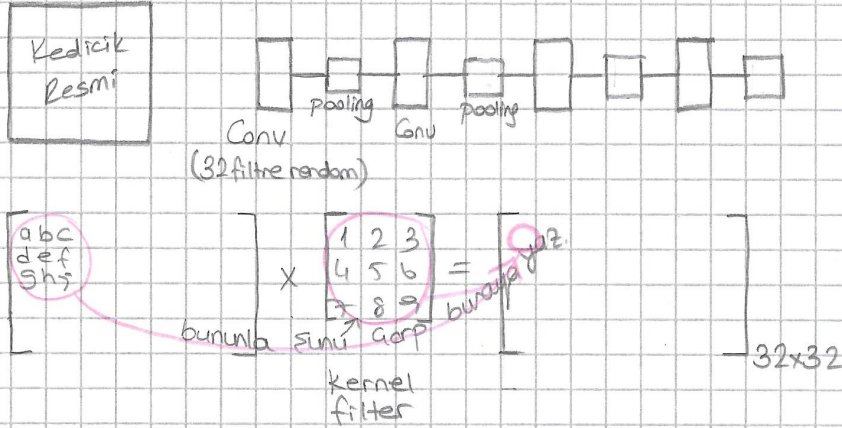


model = Sequential()

Burası convolution.

model.add(Flatten(...))

Ben bu resme;

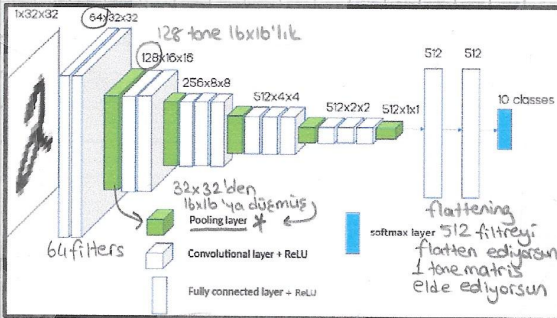


Pooling

2x2'lik pooling yaparsan yarıya düşer.

- Max Pooling : En yüksek olanı al
- Min Pooling
- Average Pooling

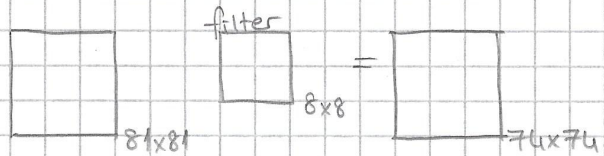
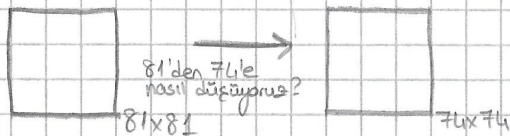
$$\begin{bmatrix} + \\ + \\ + \\ + \end{bmatrix}_{2 \times 2} \rightarrow \begin{bmatrix} - \end{bmatrix}_{1 \times 1}$$



→ pooling summarize etmeye yardımcı.
- istersen conv+conv yap, en son pooling yap.

→ Convolution'u neyle besliyoruz?
nxn resimlerle besliyoruz.

→ Kernel filtreler → random da olabilir
→ kendimizde koyabiliriz.
(Keras random oluşturuyor)

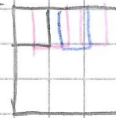


$$81 - 8 + 1 = 74$$

(original size - kernel size + 1) = ↑

* Boyutun değişmesini istemiyorsam boşlukları 0 ile doldurup yine 81x81 elde edebilirim.
74 olmasının nedeni: 81x81'e filtre uygulayınca tamamı yetmiyor.

74x74 → 2x2 pooling yap → 37x37



* Max poolingte boşlukları 0 ile doldurmanın pek anlamı yok.

3x3 pooling
yaptığını
düşün

0	0	0	0	0	0
0	0	0	0	0	0
0	0	5	0	0	0
0	0	4	1	2	0

→ 0 ile doldursanda max değer 5 gelecek.