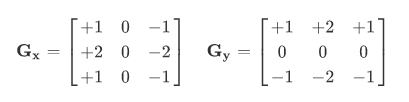
1.

a)

i)



ii)

f\*hx=

|  |  |  |
| --- | --- | --- |
| 8 | 8 | 8 |
| 8 | 8 | 8 |
| 8 | 8 | 8 |

f\*hy=

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

b)

i)

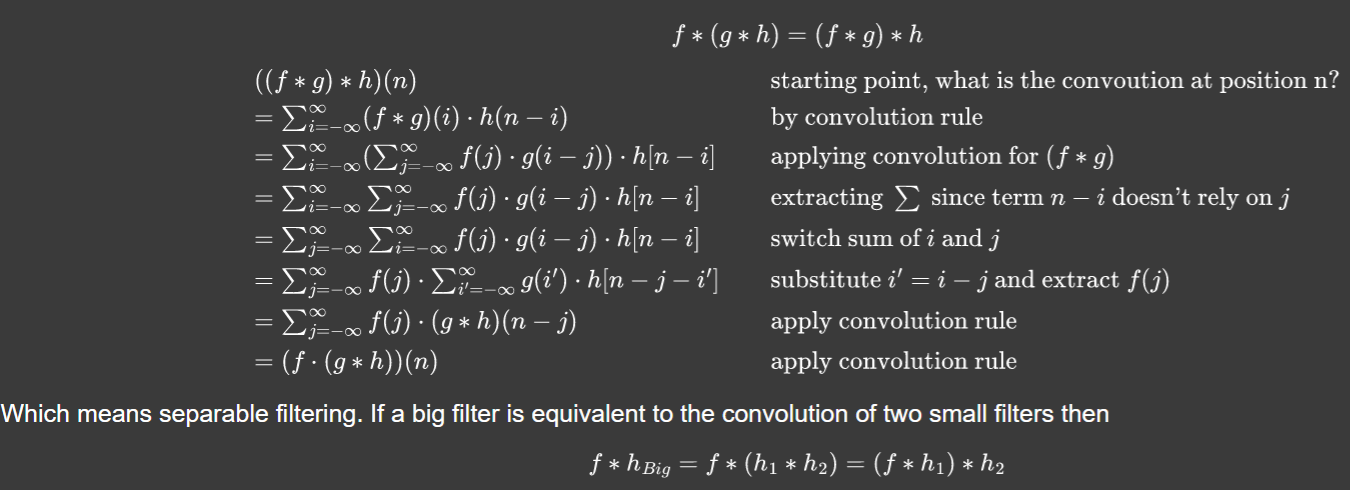
To smooth the image, because edge detection is sensitive to noise.

Provide information at different scales.

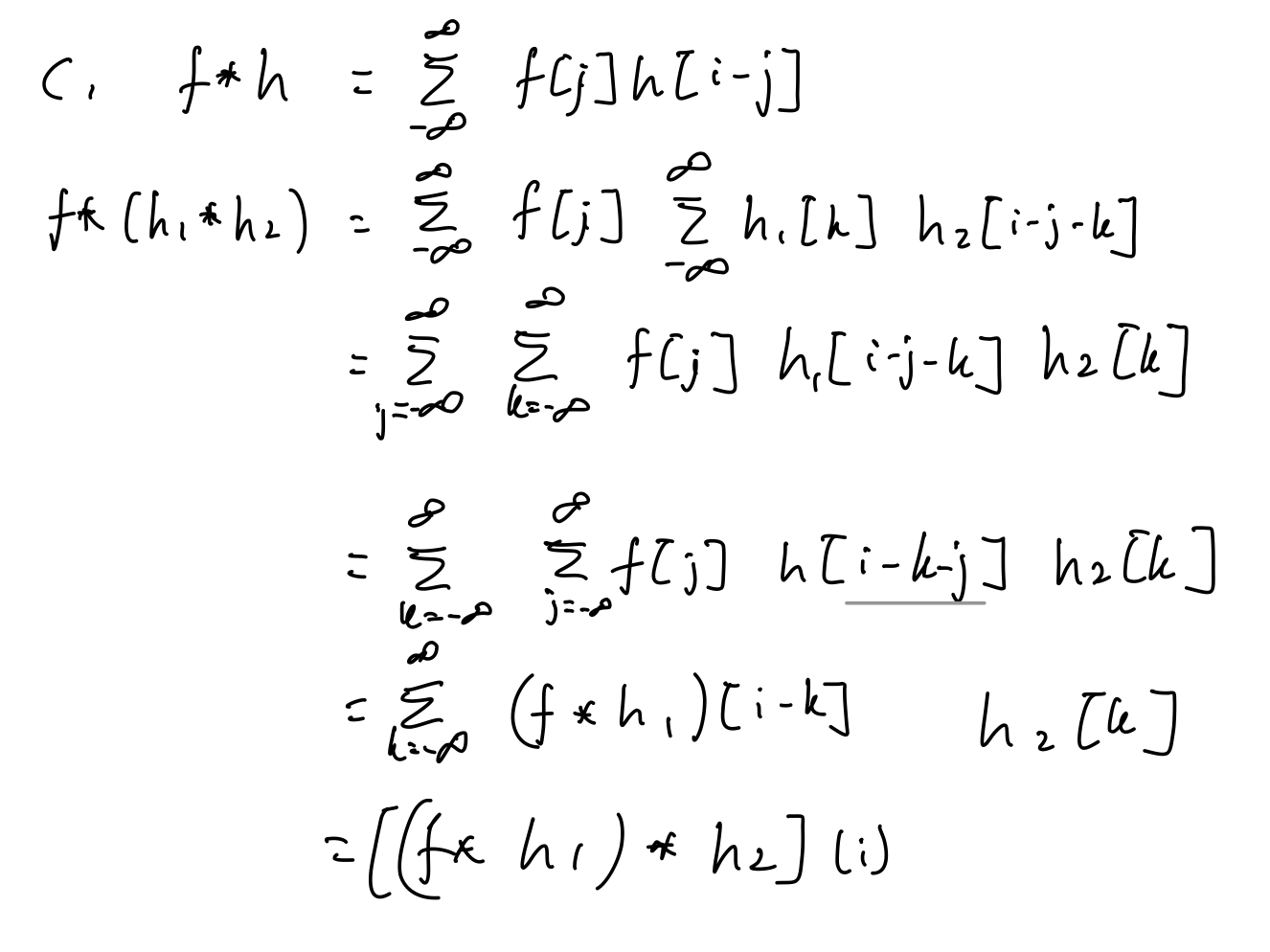
ii)

O(N^2 K)

c)

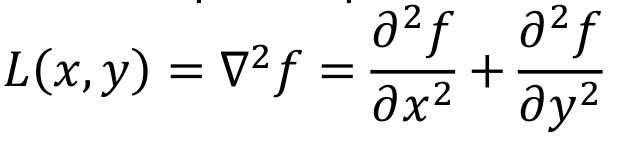


Alternatively, without the substitution:



2.

a)i)



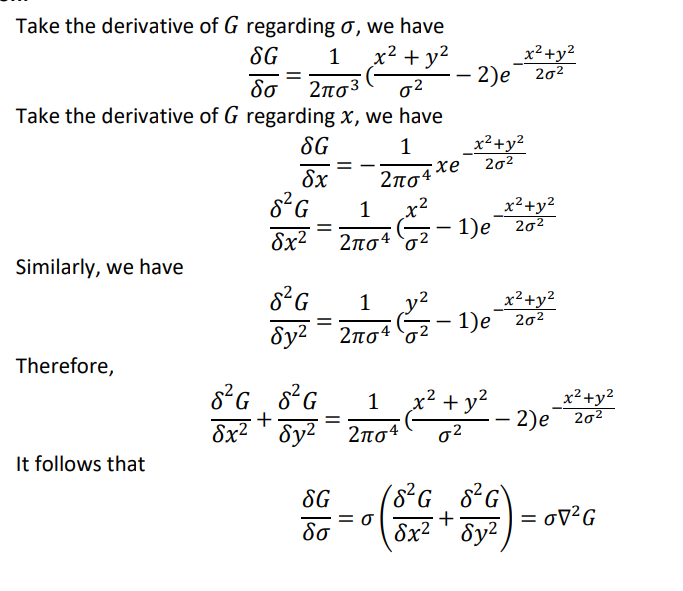
ii)

image\*filter=

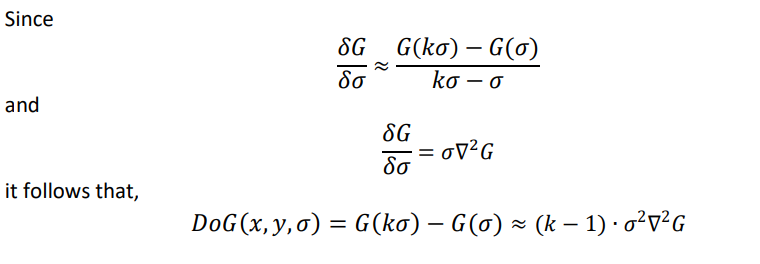
|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

iii) A gaussian filter is applied first to smoothen out the noise, followed by a laplace filter for detection.

b) i) Copied from Tutorial 4 answers:



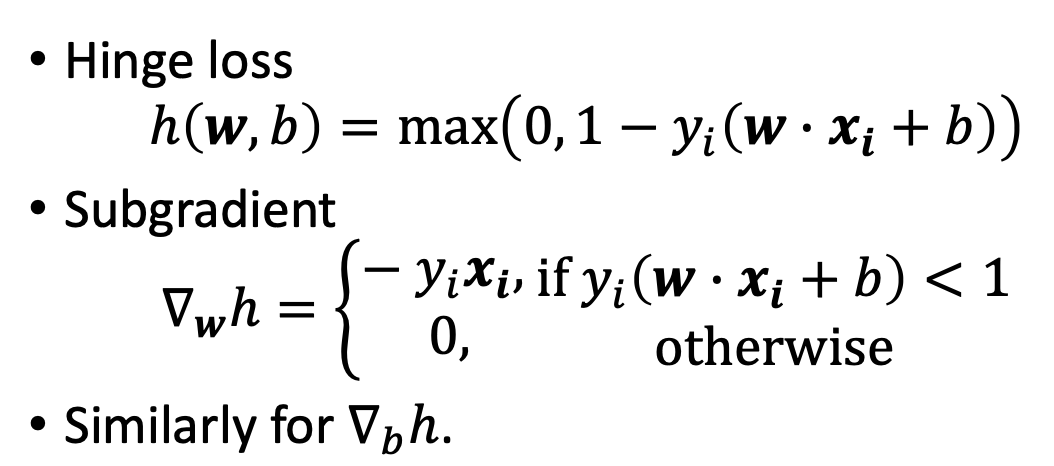
ii) Copied from tutorial 4 answers



3)

i) All test samples are X, since they are 1 euclidean away from X and sqrt(2) from O

ii) All test samples are now O, since the 3 closest are always 2 O, 1 X

iii) B No training cost and high test cost

b) i)

I’m likely wrong but I think

dh/dw = -y\_i\*x\_i if y\_i(w\*x\_i + b) < 1, 0 otherwise

dh/db = -y\_i if y\_i(w\*x\_i + b) < 1, 0 otherwise

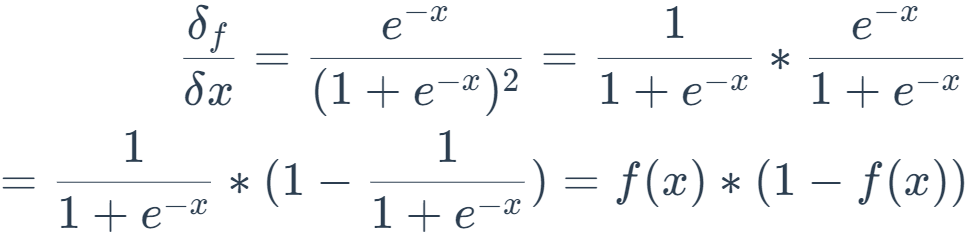
ii) You would have to build m(m-1)/2 classifiers, one for each possible pair. The classification result would be chosen by the class that “wins” the most classifiers.

iii) With M classes in one-vs-all, we would need M classifiers. The classifier with the highest response determines the result./

4)

a)

i)



ii) The problem with sigmoid in gradient descent is that at small and large values (of x that result in f(x) close to 0 or 1), the gradient becomes very small, so that during propagation the values do not change very much, causing it to learn very slowly.

iii) The gradient of ReLU is 1 if x >= 0, 0 if x < 0

iv) When x is negative, the ReLU output and its derivative are both 0, meaning that nothing is changed during gradient descent.

b)

i) We would need NxNxM connections

ii) It would be 9 x number of filters. (M?)

9xN

xN?

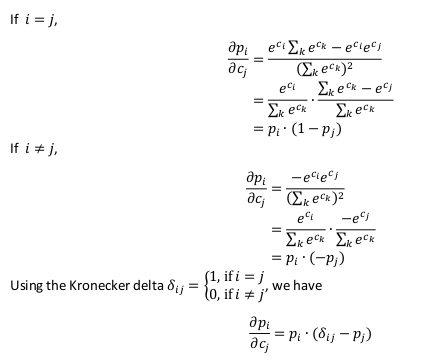
9xC where C = no. of channels? -> greyscale image so C = 1 thus 9 connection weights?

iii) A) Translation invariant

iv) If i = j, then we get p(i)(1-p(i))

If i =/= j, then we get - p(i) \* p(j)

Derivations are left to the reader



(From Tutorial 7, Question 3)

v) Cross entropy

In short, cross-entropy(CE) is the measure of how far is your predicted value from the true label.

The cross here refers to calculating the entropy between two or more features / true labels (like 0, 1).

And the term entropy itself refers to randomness, so large value of it means your prediction is far off from real labels.