Title: Machine Learning Lecture

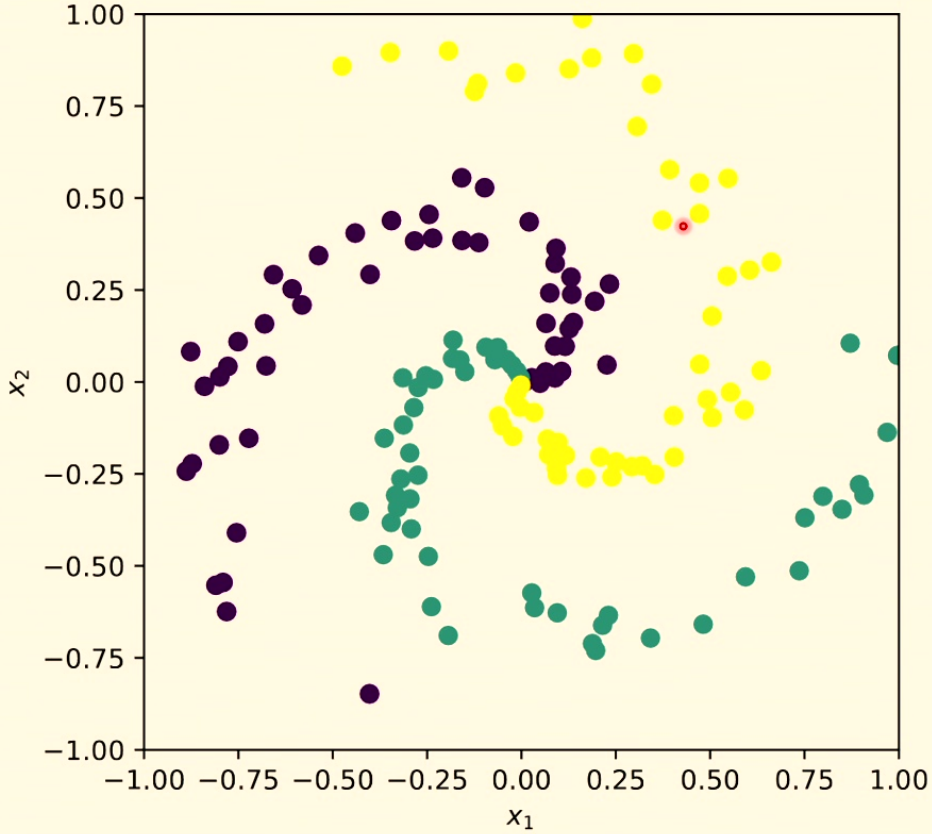
Speakers: Mohamed Hibat Allah

Collection: Machine Learning 2023/24

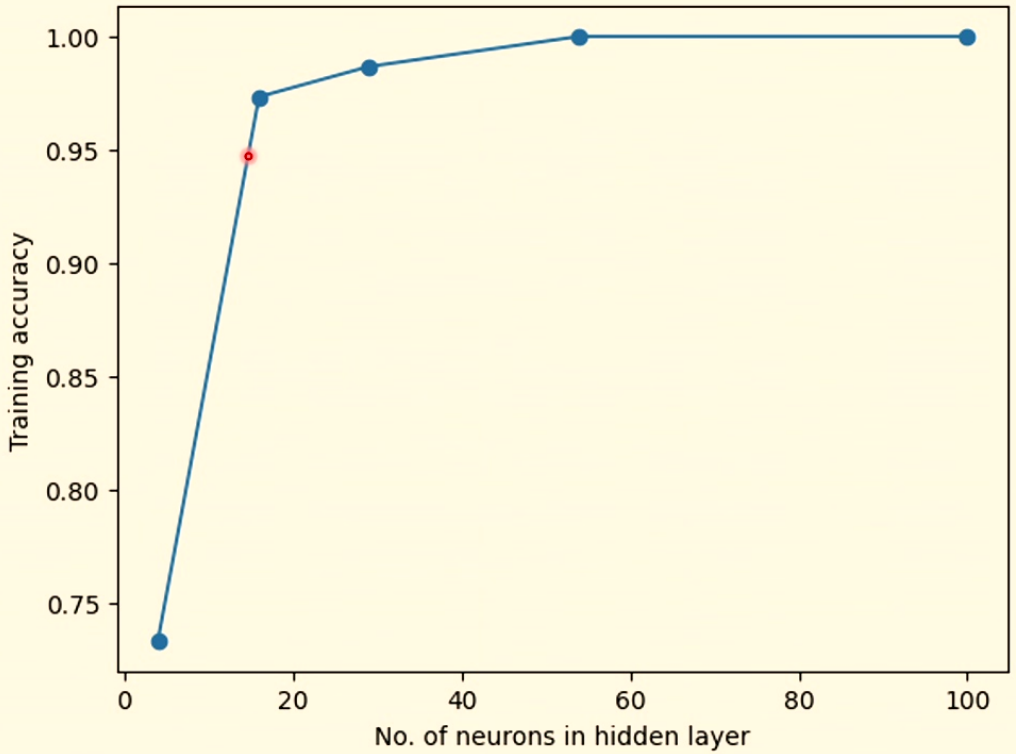
Date: April 18, 2024 - 11:30 AM

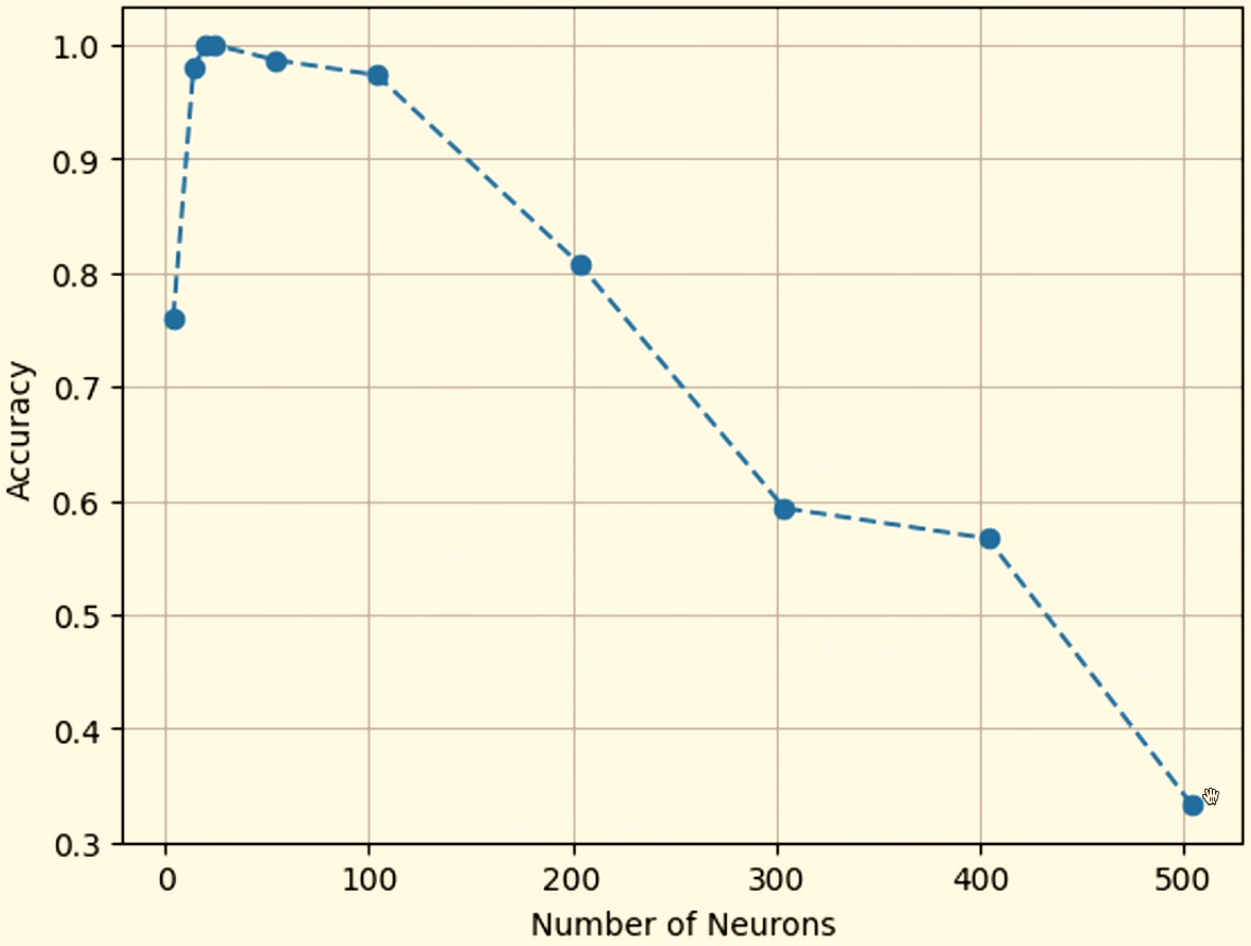
URL: https://pirsa.org/24040051

Tutorial 3

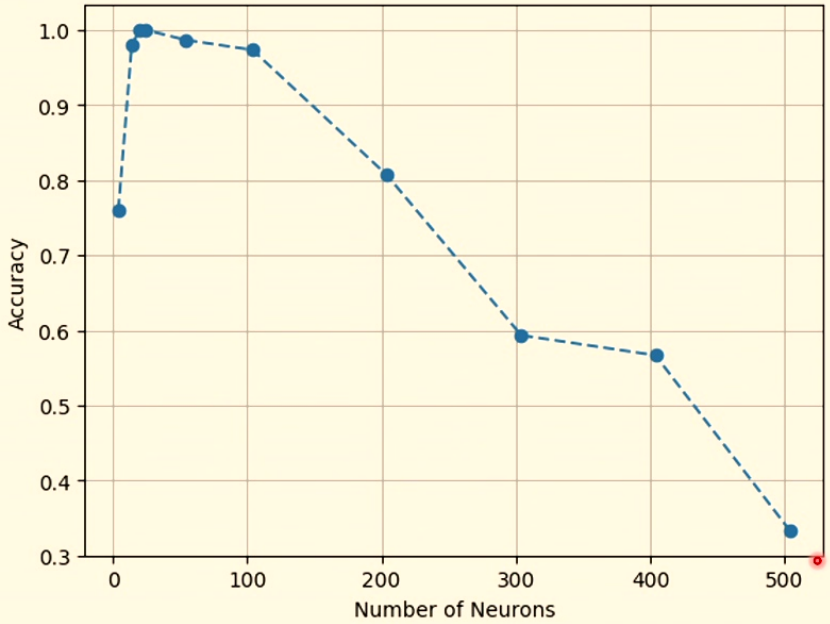

1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x_2
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00
x_1

Tutorial 3 results

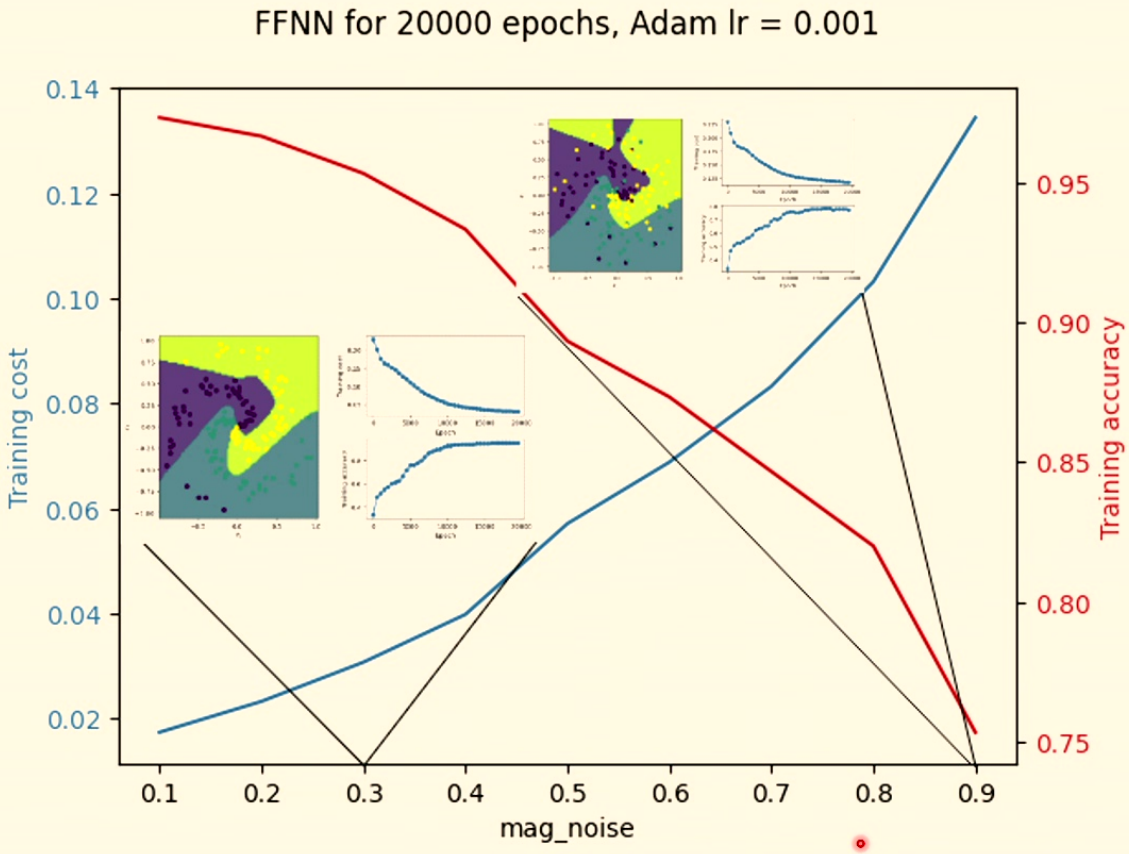



1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
Accuracy
0
100
200
300
400
500
Number of Neurons

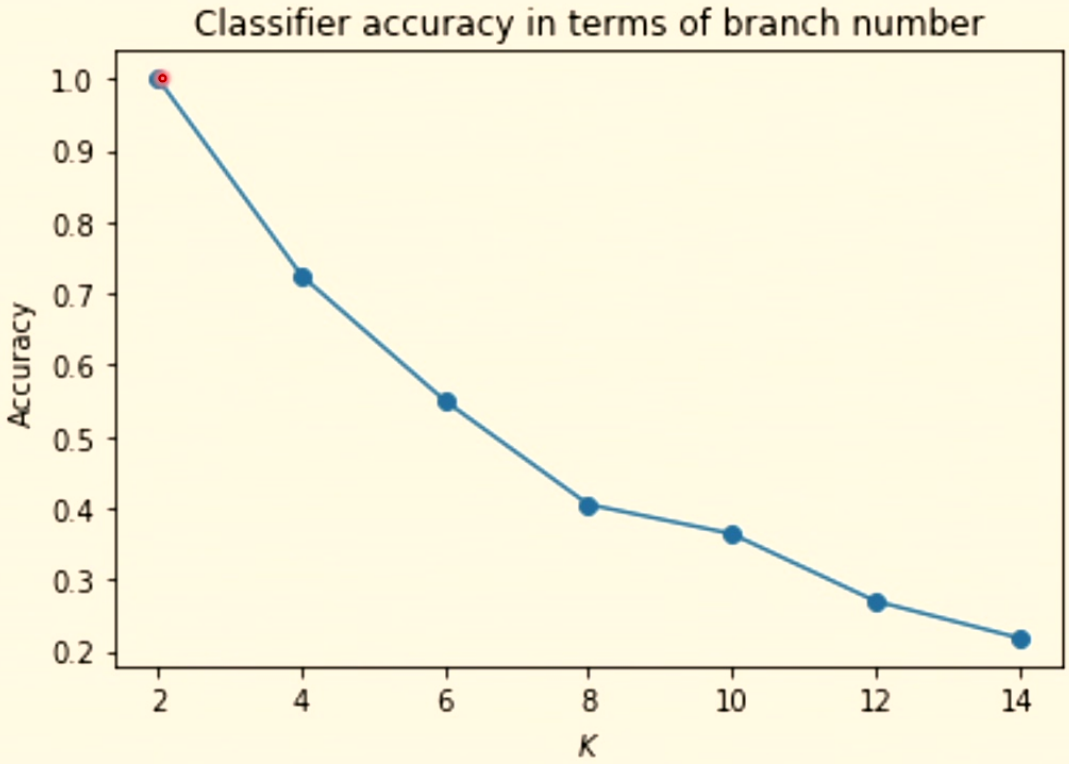
Tutorial 3 results



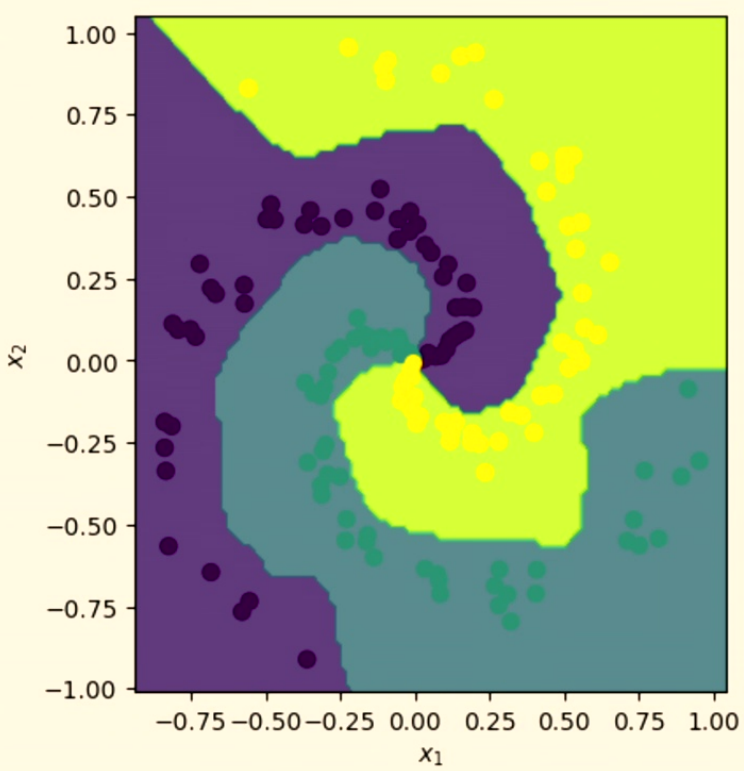
Tutorial 3 results



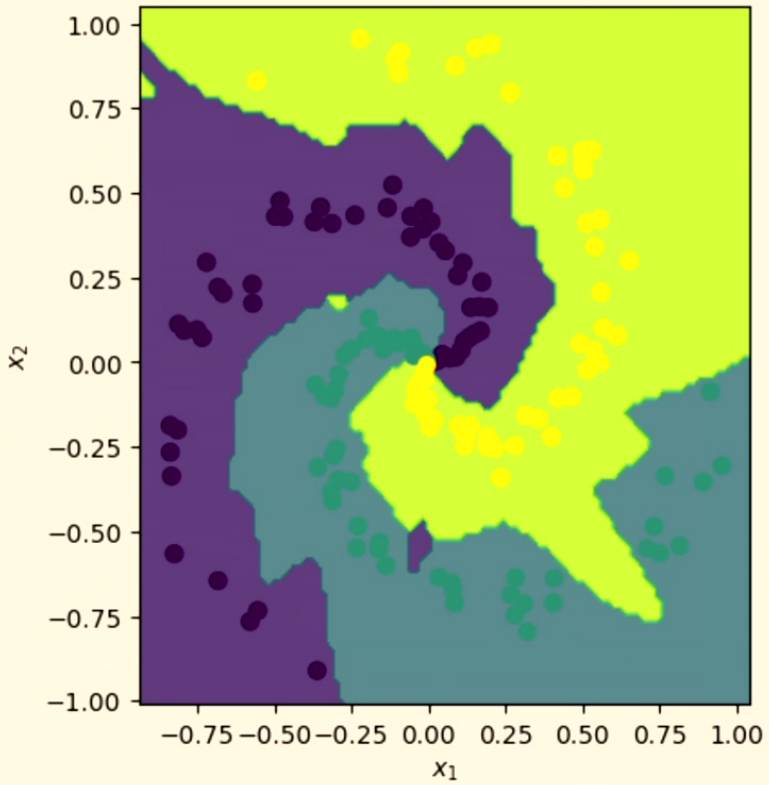
Tutorial 3 results

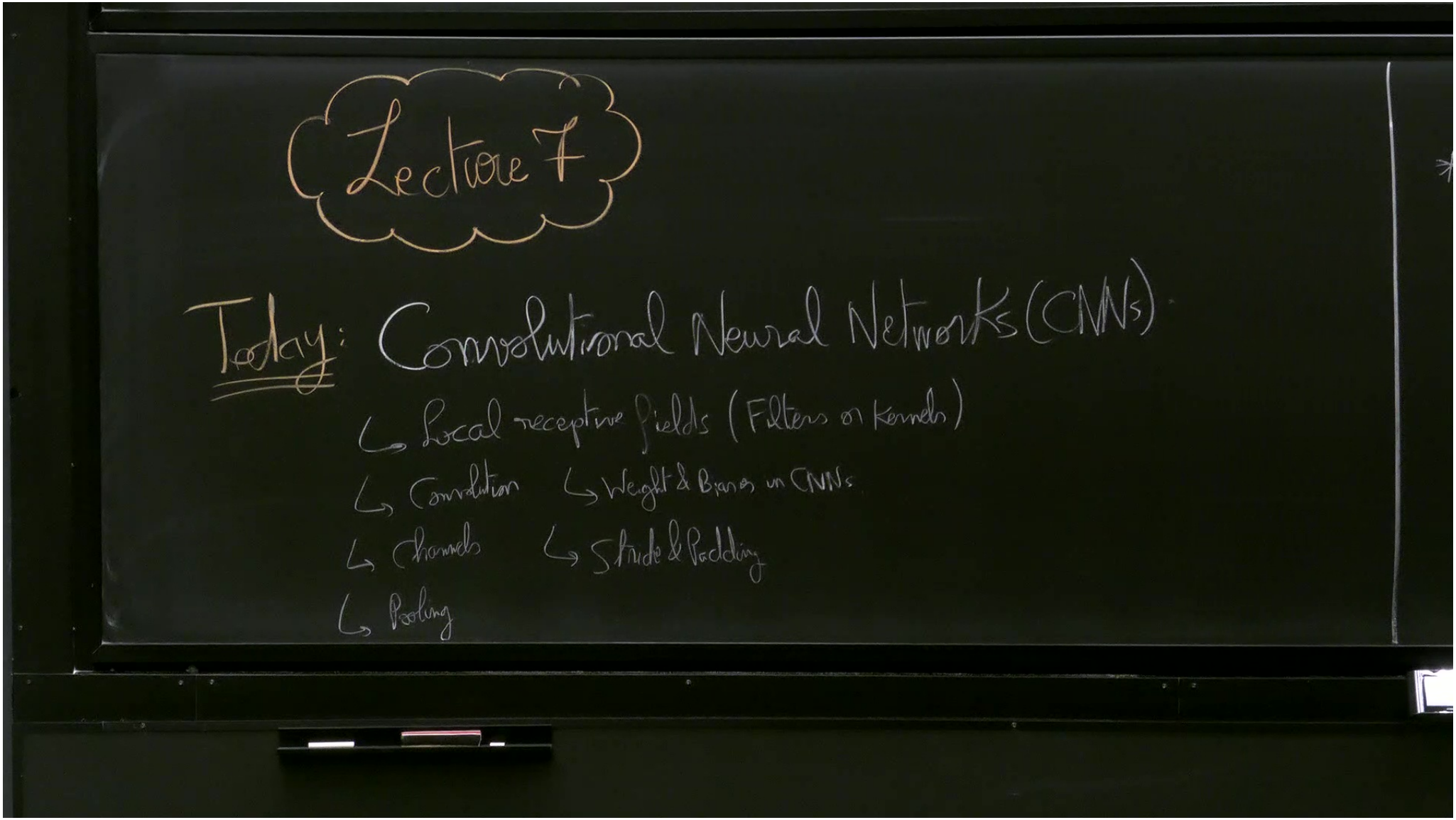


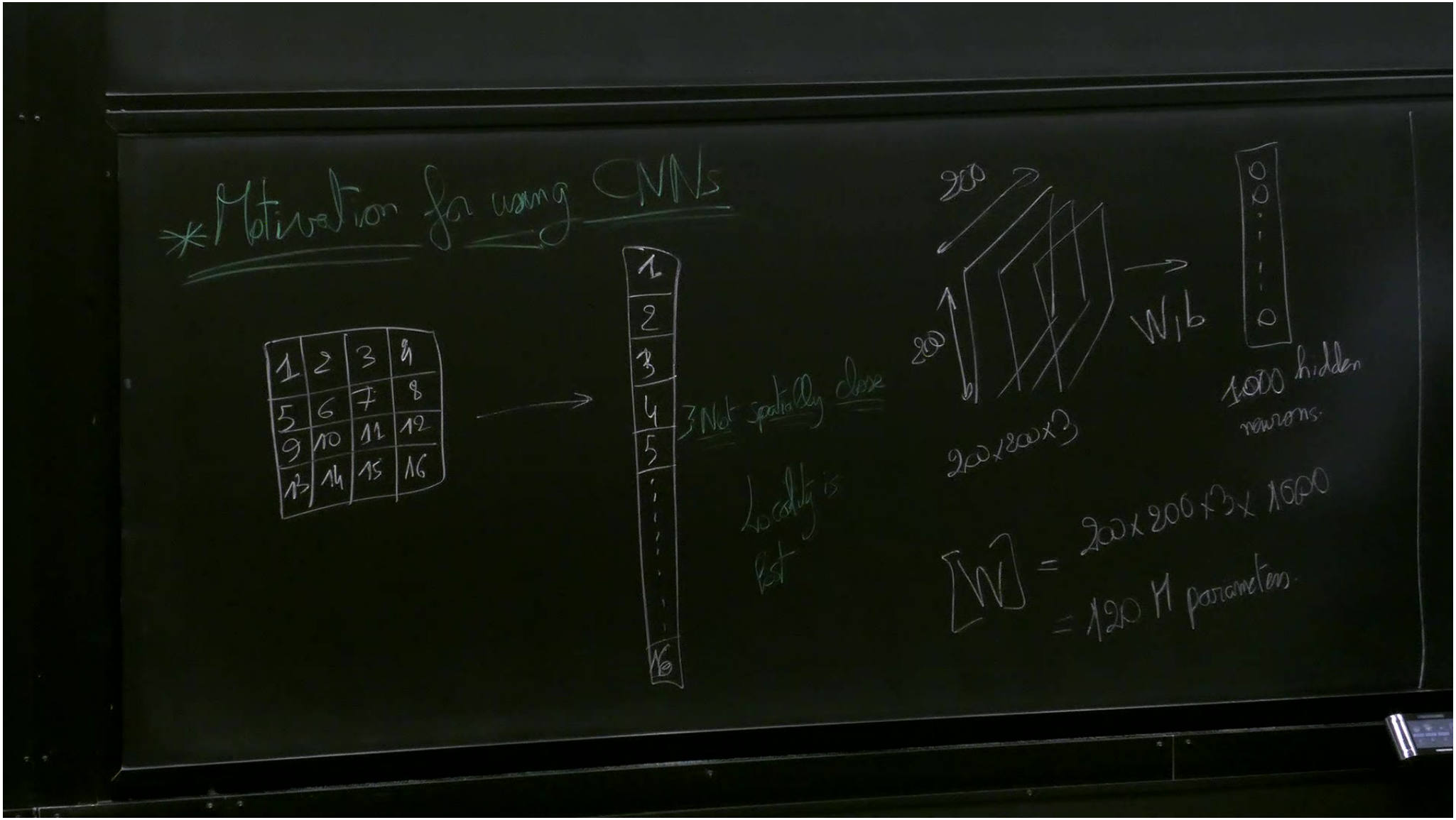
Tutorial 3 results

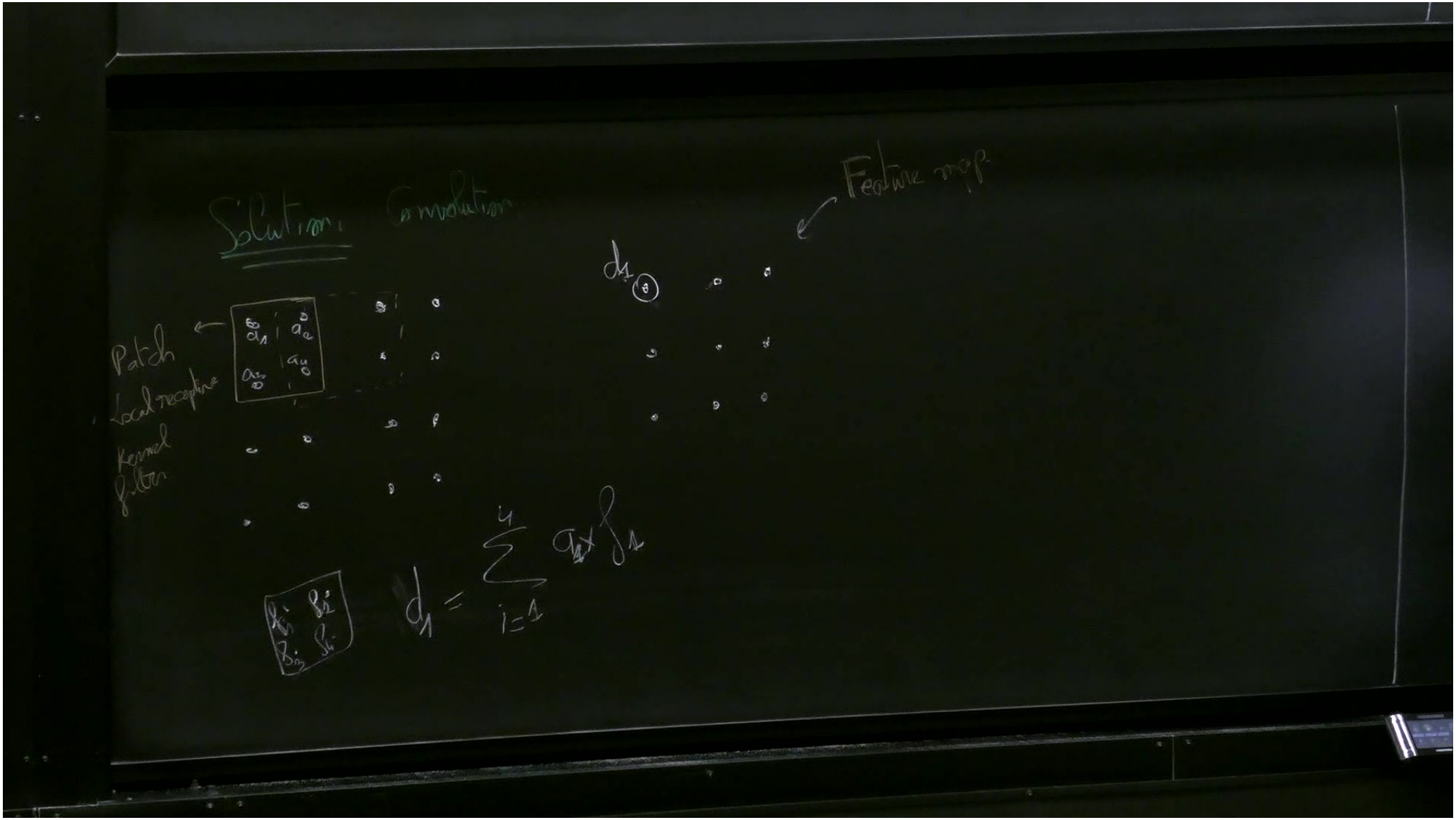


Early stopping




Lecture 7
Today: Convolutional Neural Networks (CNNs)
↳ Local receptive fields (Filters or Kernels)
↳ Convolution
↳ Weight & Biases in CNNs
↳ Channels
↳ Stride & Padding
↳ Pooling


*Motivation for using CNNs
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Not spatially close
Locality is lost
200
200
200x200x3
W,b
1000 hidden neurons.
[W] = 200x200x3x1000
= 120 M parameters.


Solution: Convolution
Feature map
Patch
Local receptive
Kernel
filter
$d_1 = \sum_{i=1}^{4} a_i * f_1$

master
Go to file t
bin
gif
.gitignore
arbitrary_padding_no_strides.gif
arbitrary_padding_no_strides_tr...
dilation.gif
full_padding_no_strides.gif
full_padding_no_strides_transp...
no_padding_no_strides.gif
no_padding_no_strides_transpo...
no_padding_strides.gif
no_padding_strides_transposed...
padding_strides.gif
padding_strides_odd.gif
padding_strides_odd_transpose...
padding_strides_transposed.gif
same_padding_no_strides.gif
same_padding_no_strides_trans...
natbib
pdf
png
templates
.gitignore

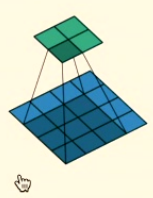
A technical report on convolution arithmetic in the context of deep learning.

The code and the images of this tutorial are free to use as regulated by the licence and subject to proper attribution:

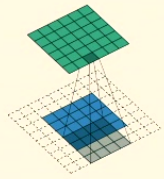
- [1] Vincent Dumoulin, Francesco Visin - A guide to convolution arithmetic for deep learning (BibTeX)

Convolution animations

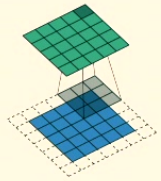
N.B.: Blue maps are inputs, and cyan maps are outputs.



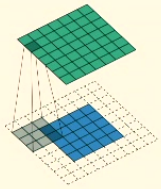
No padding, no strides



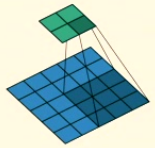
Arbitrary padding, no strides



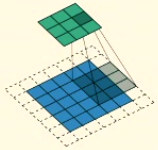
Half padding, no strides



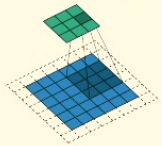
Full padding, no strides



No padding, strides



Padding, strides



Padding, strides (odd)

Transposed convolution animations

N.B.: Blue maps are inputs, and cyan maps are outputs.

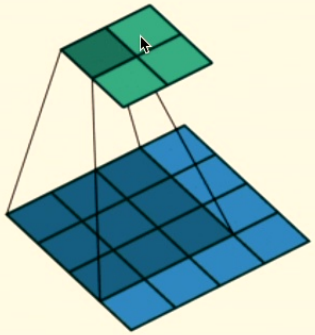
Firefox File Edit View History Bookmarks Tools Window Help

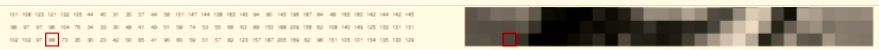
zoom Thu Apr 18 11:48 AM

NN SVG | conv_arithmetic/gif/no_padding | Image Kernels explained visually | CNN Explainer | Machine Learning for Many-Bo | PSI Portal

https://github.com/vdumoulin/conv_arithmetic/blob/master/gif/no_padding_no_strides.gif

36.9 KB Code 55% faster with GitHub Copilot



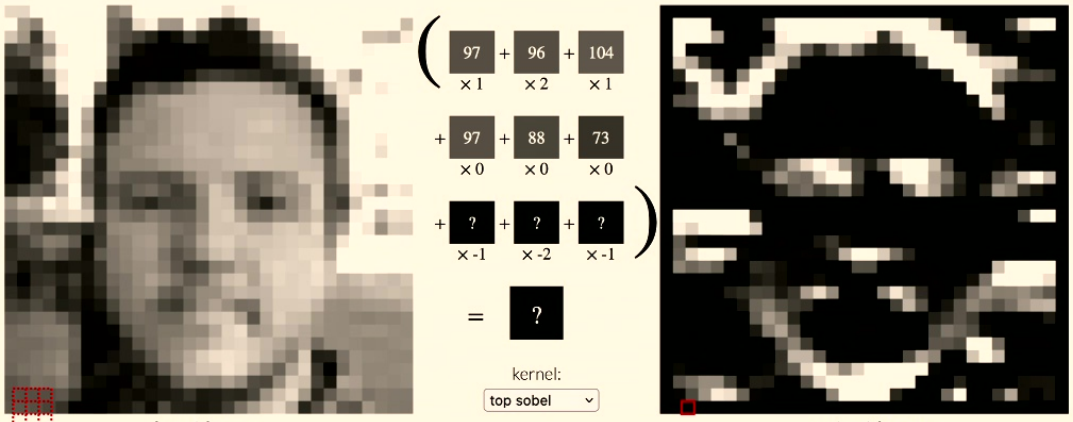


Let's walk through applying the following 3x3 **top sobel** kernel to the image of a face from above.

top sobel

$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

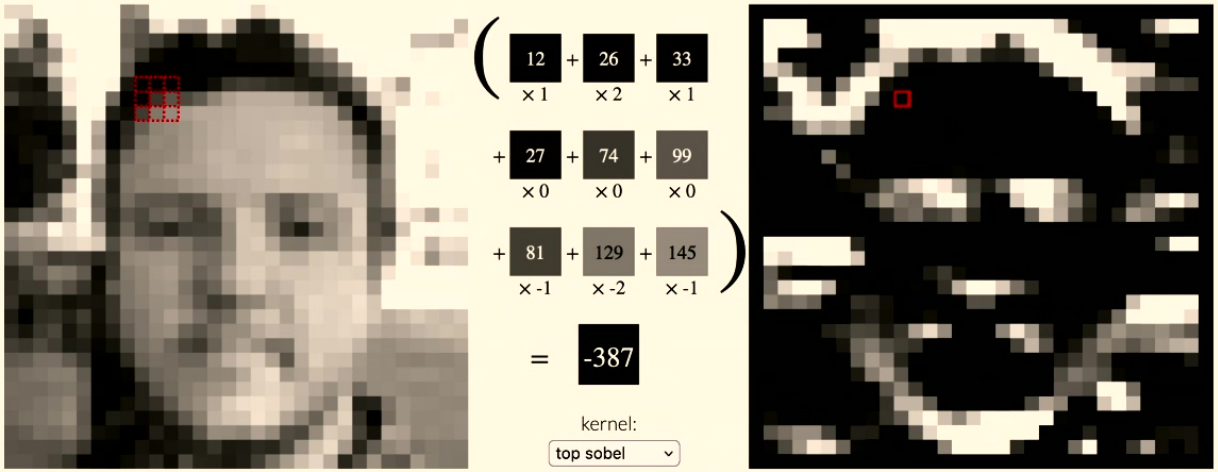
Below, for each 3x3 block of pixels in the image on the left, we multiply each pixel by the corresponding entry of the kernel and then take the sum. That sum becomes a new pixel in the image on the right. Hover over a pixel on either image to see how its value is computed.



One subtlety of this process is what to do along the edges of the image. For example, the top left corner of the input image only has three neighbors. One way to fix this is to extend the edge values out by one in the original image while keeping our new image the

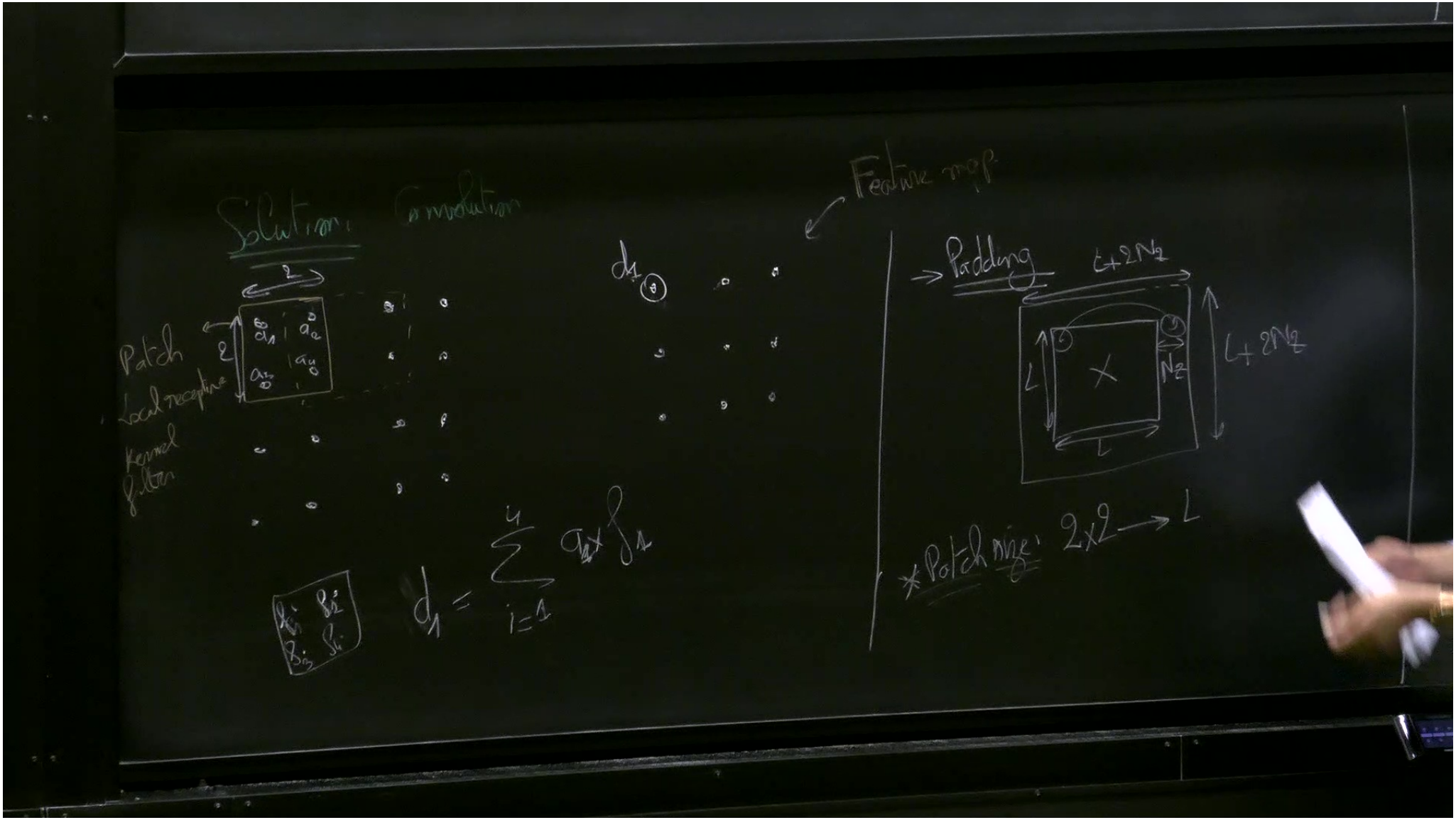
$$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

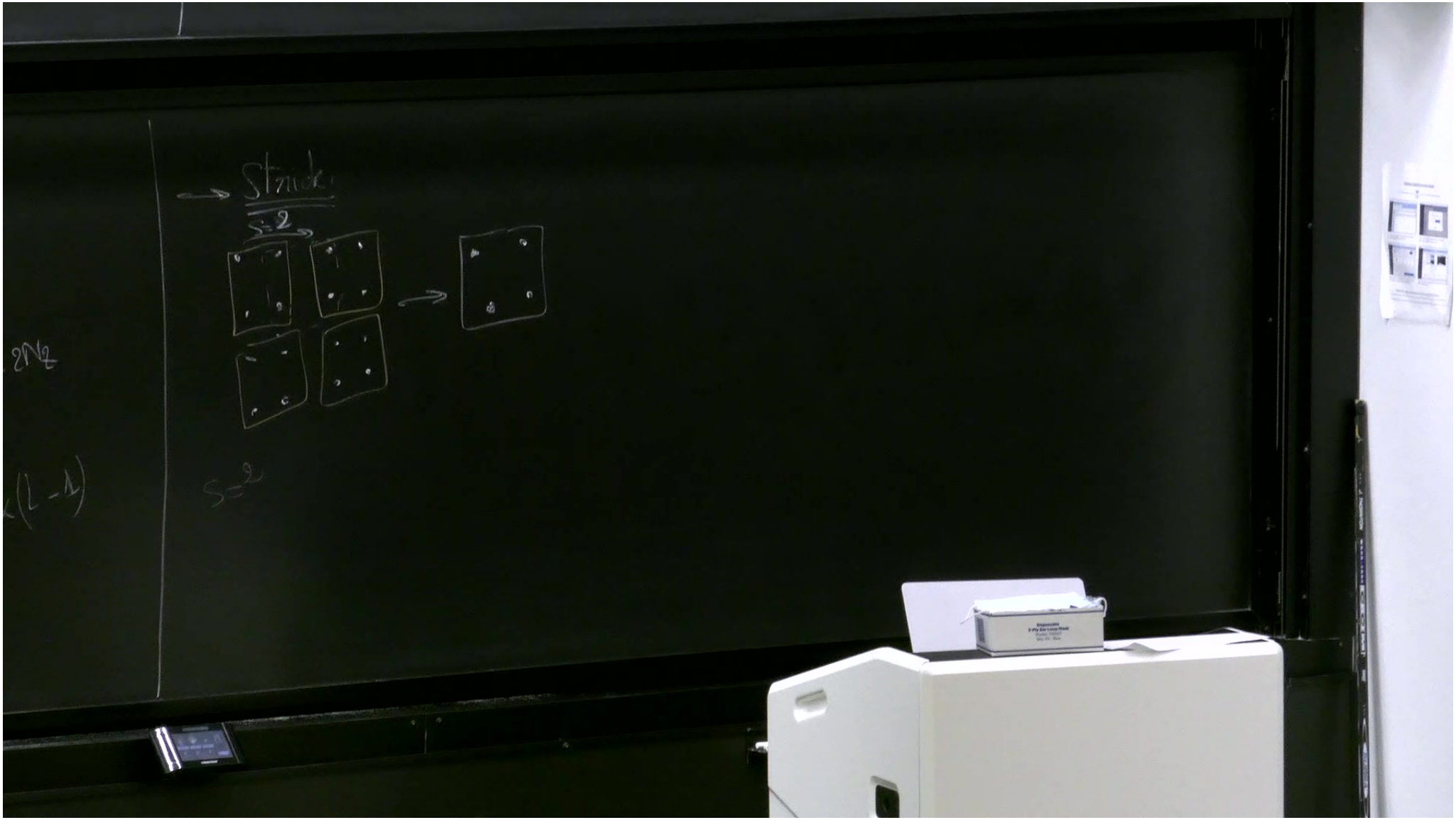
Below, for each 3x3 block of pixels in the image on the left, we multiply each pixel by the corresponding entry of the kernel and then take the sum. That sum becomes a new pixel in the image on the right. Hover over a pixel on either image to see how its value is computed.



input image output image

One subtlety of this process is what to do along the edges of the image. For example, the top left corner of the input image only has three neighbors. One way to fix this is to extend the edge values out by one in the original image while keeping our new image the same size. In this demo, we've instead ignored those values by making them black.


Solution: Convolution
Feature map
Patch
Local receptive
Kernel
filter
Padding
L+2N2
L+2N2
N2
L
L
X
* Patch size: 2x2 → L

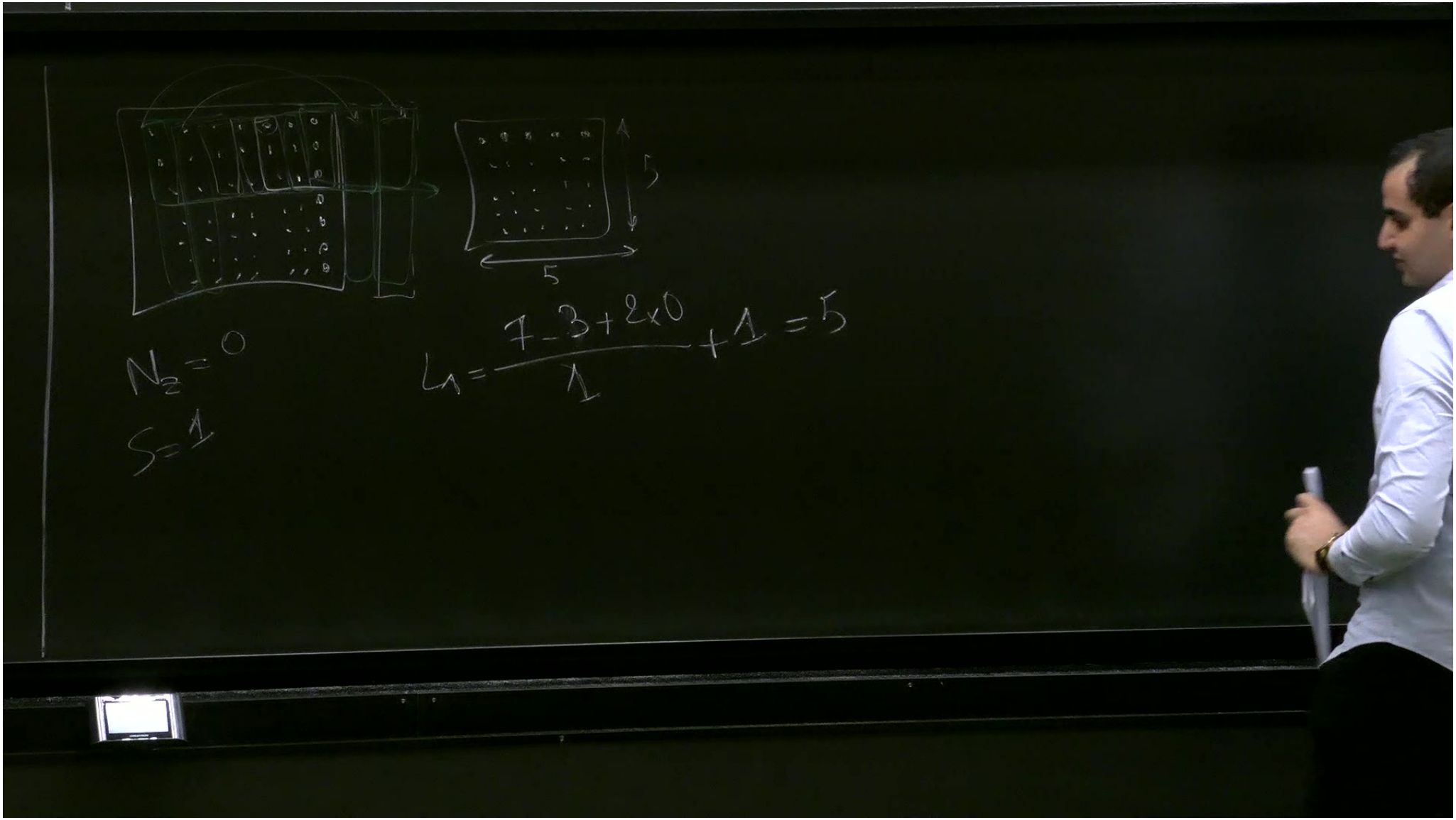

Stride
s=2
s=2

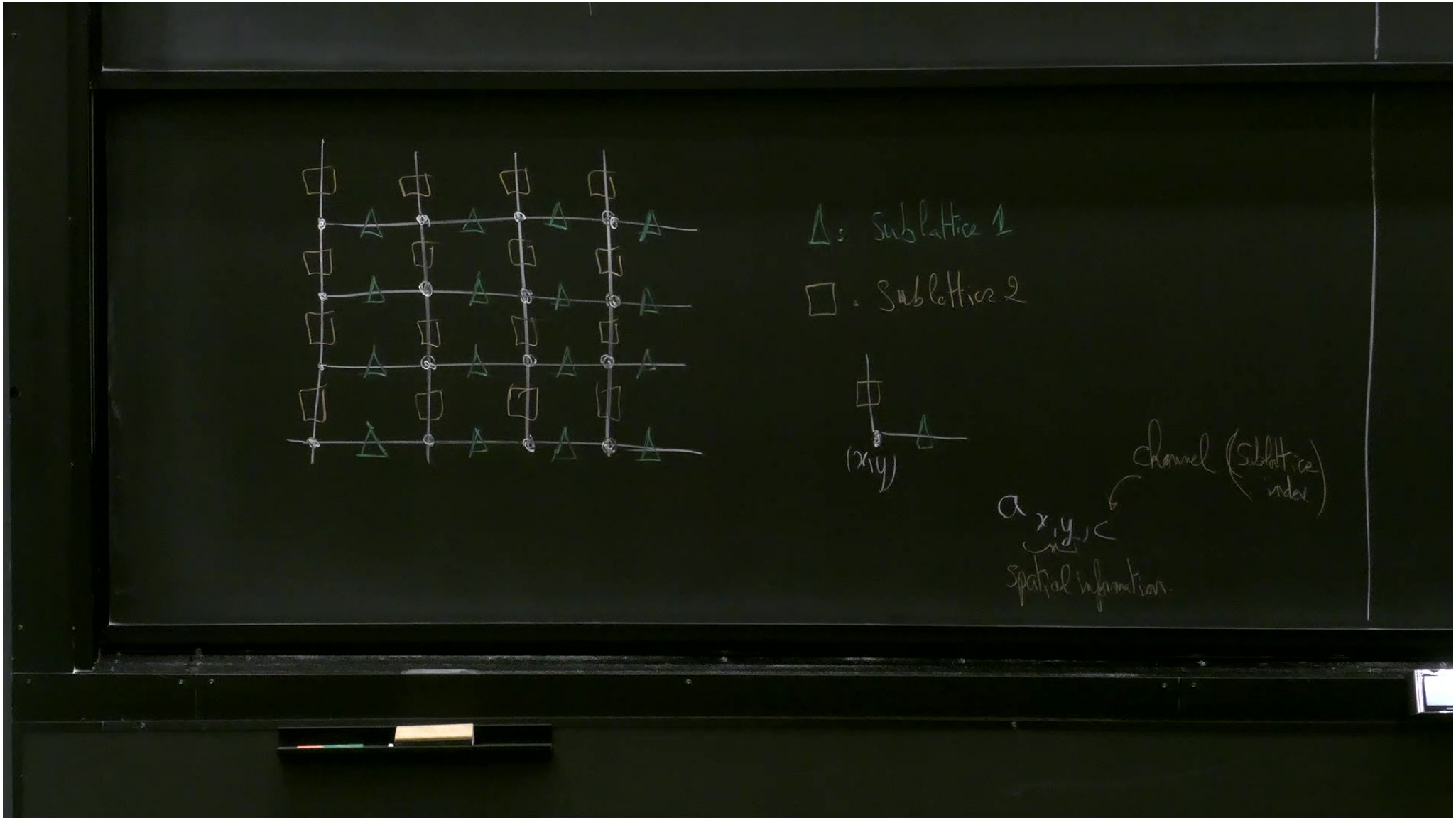
$x = \Omega^{(0)} \longrightarrow L \times L \longrightarrow a^{(1)} \longrightarrow L_1 \times L_1$

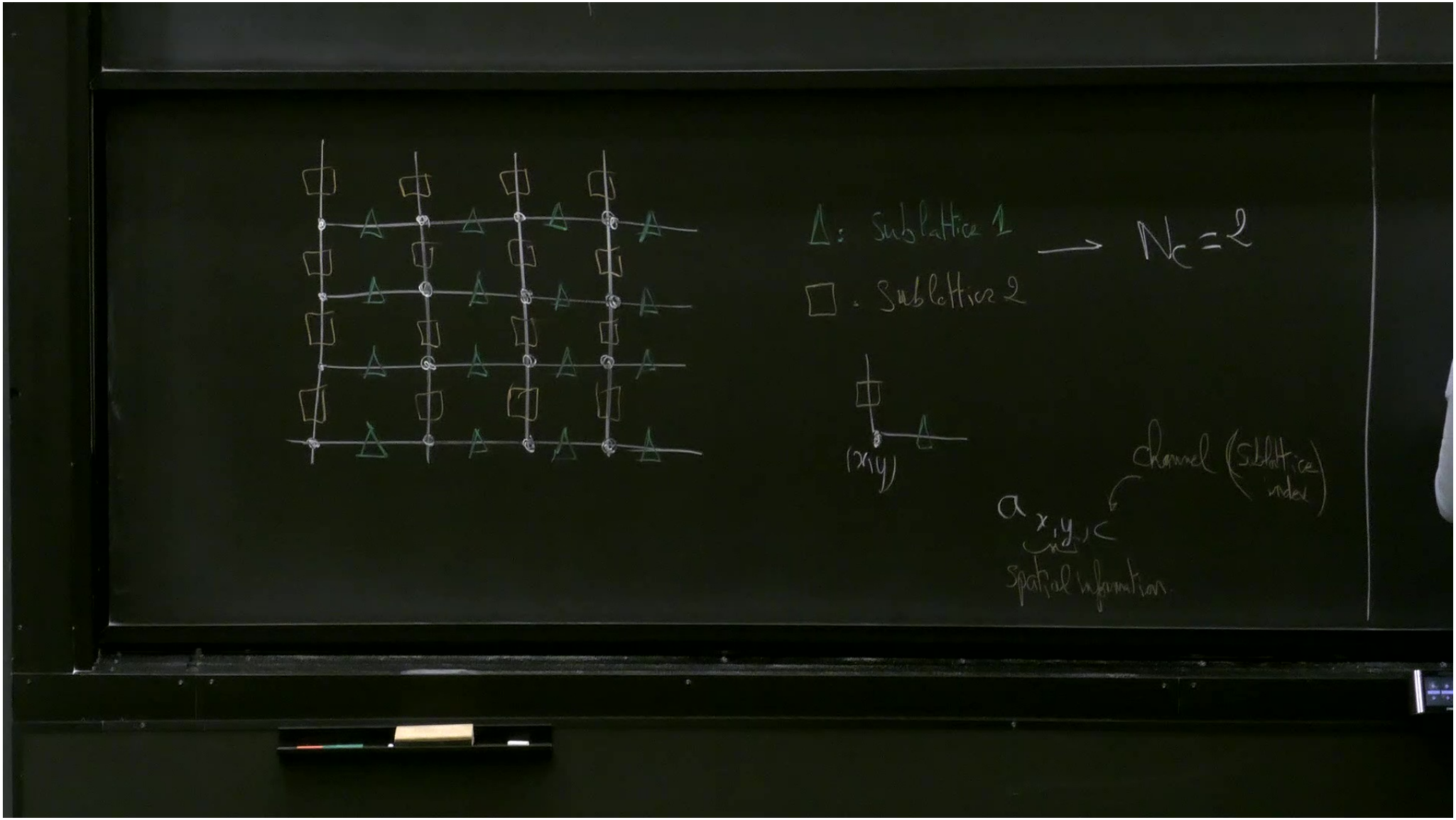
$$L_1 = \frac{L - L_{patch} + 2N_z}{s} + 1 \longrightarrow \text{has to be an integer.}$$

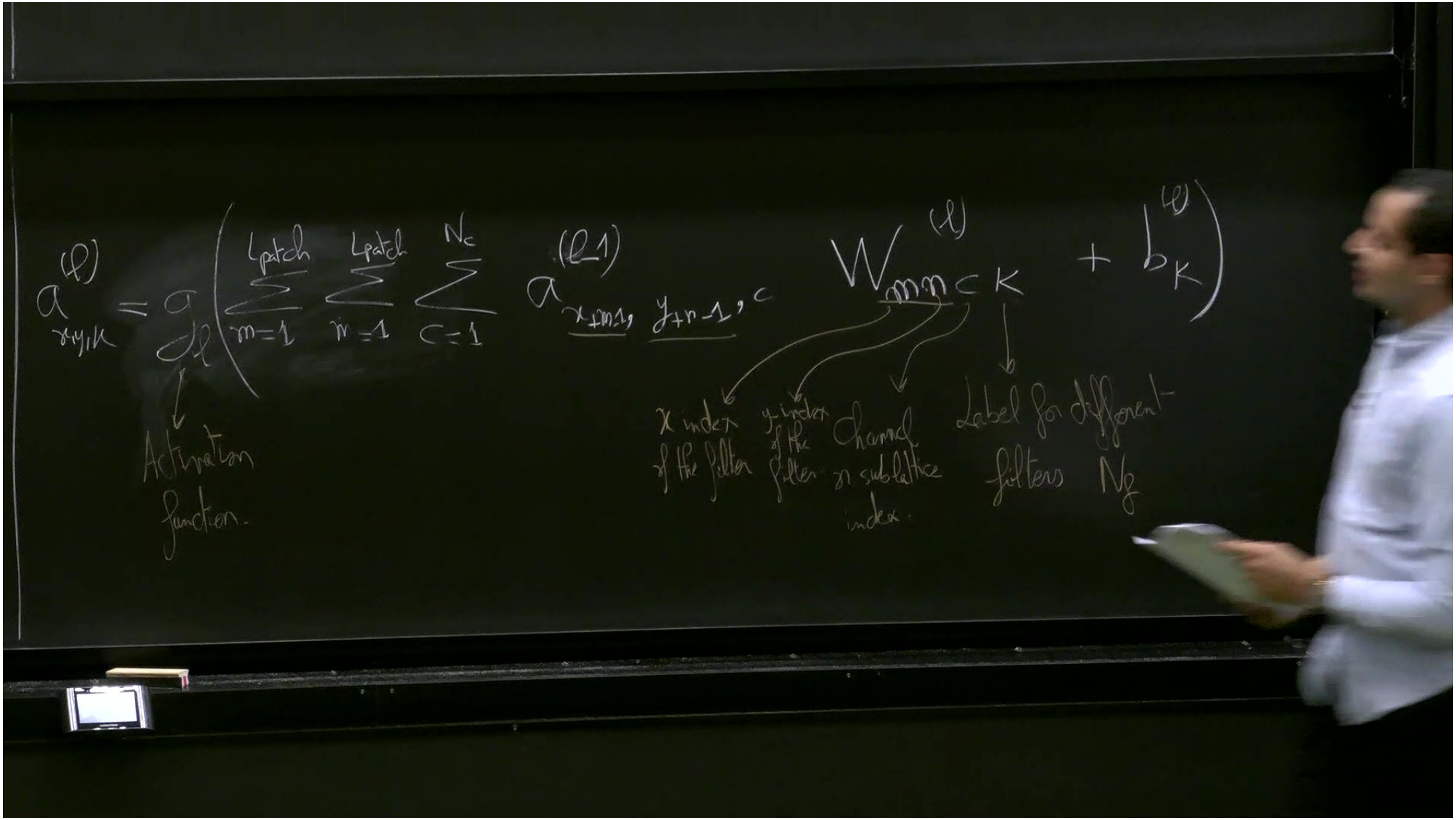
Using periodic boundary conditions.

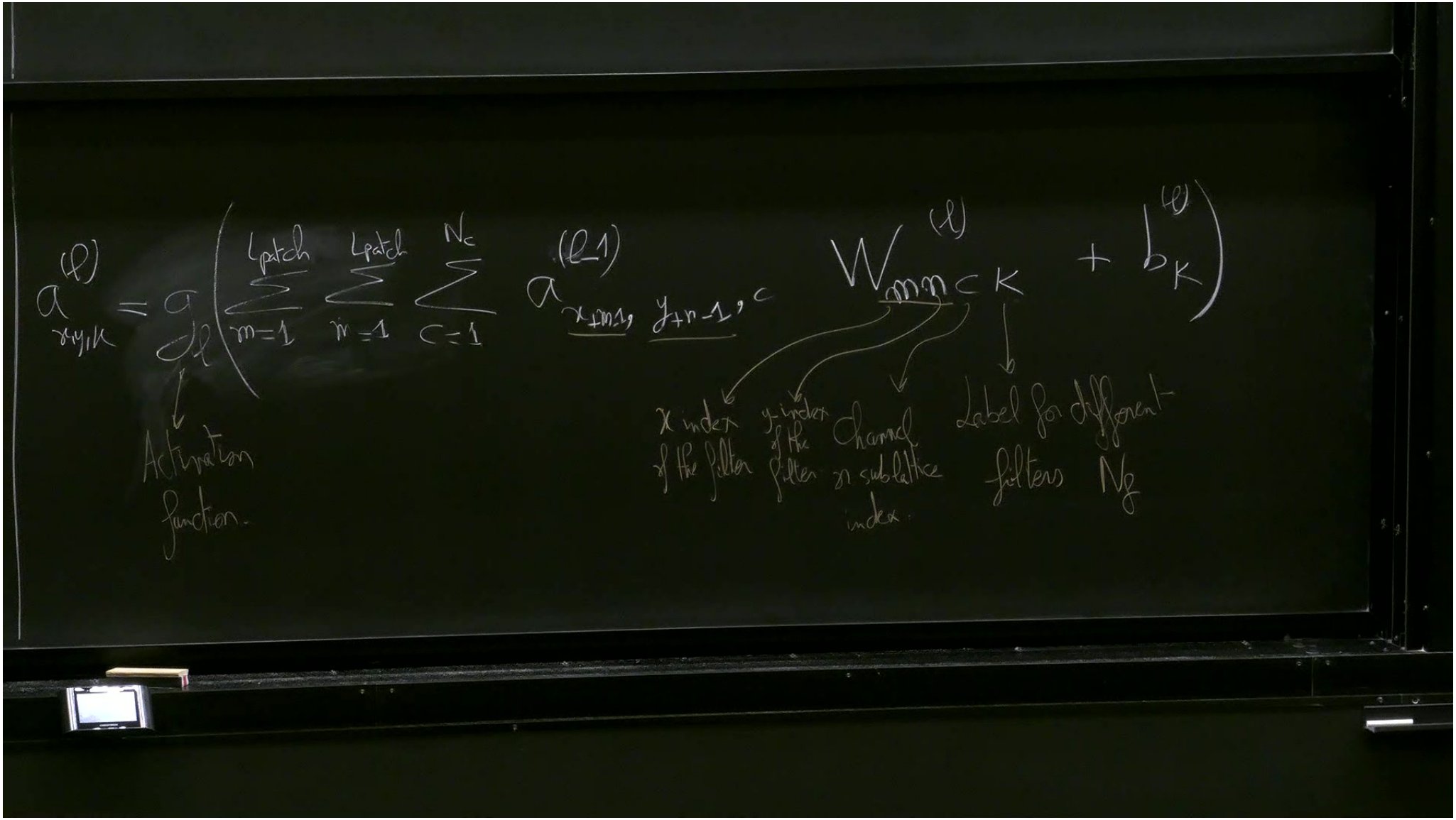
$$L_1 = \frac{L}{s}$$

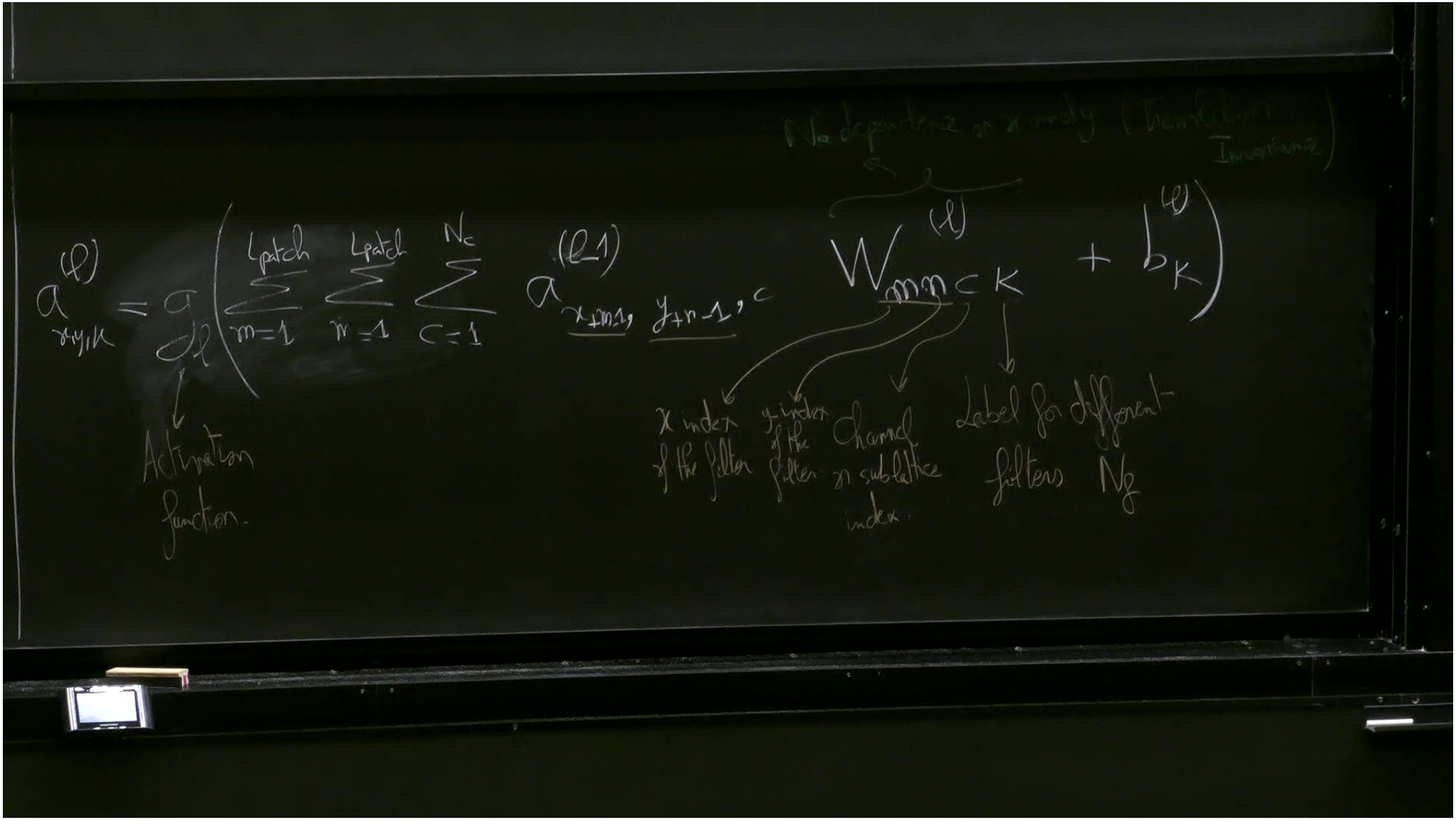

5
5
$N_2 = 0$
$S = 1$
$L_1 = \frac{7 - 3 + 2 \times 0}{1} + 1 = 5$

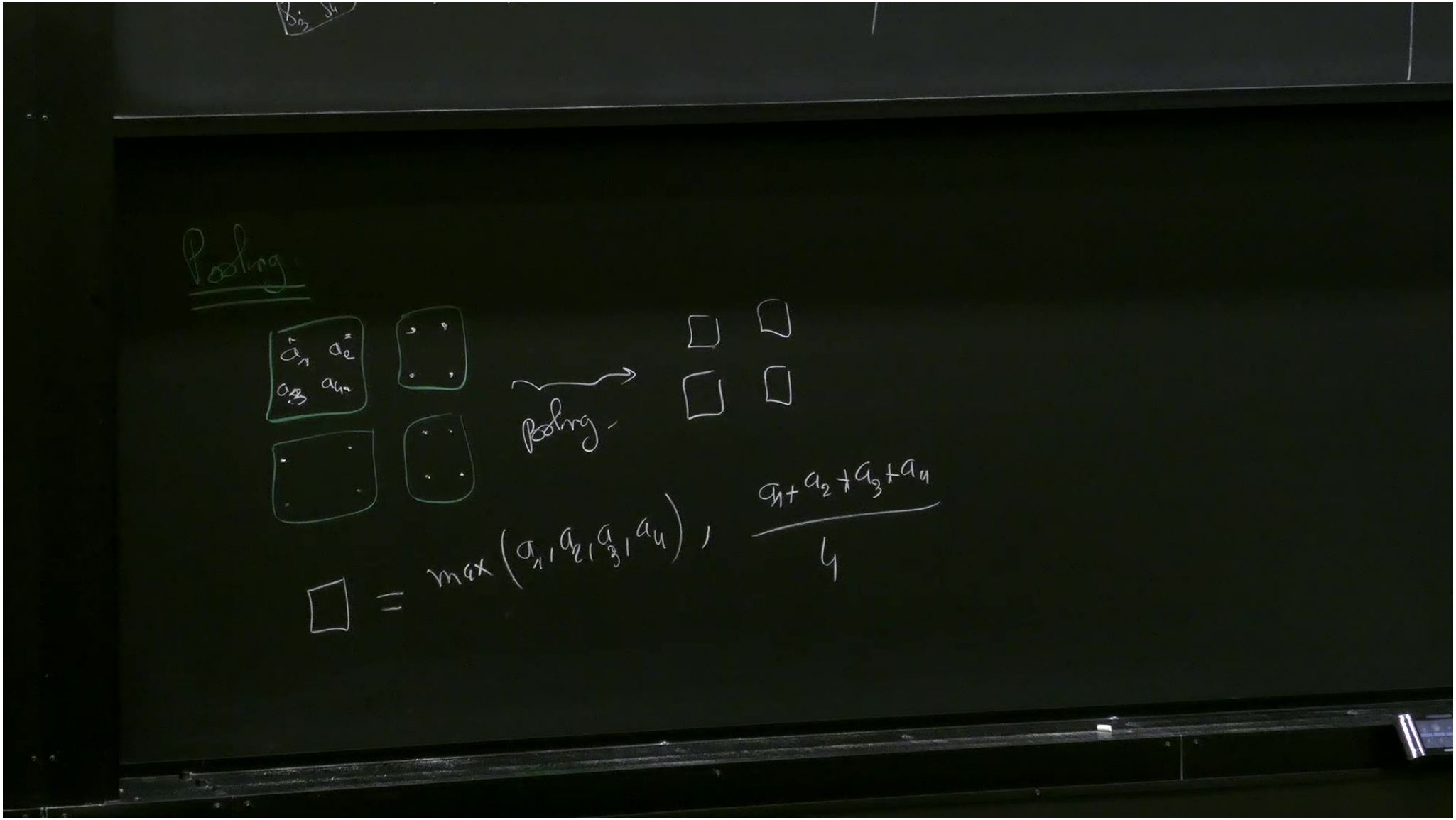

Δ: Sublattice 1
□: Sublattice 2
(x, y)
channel (Sublattice index)
spatial information

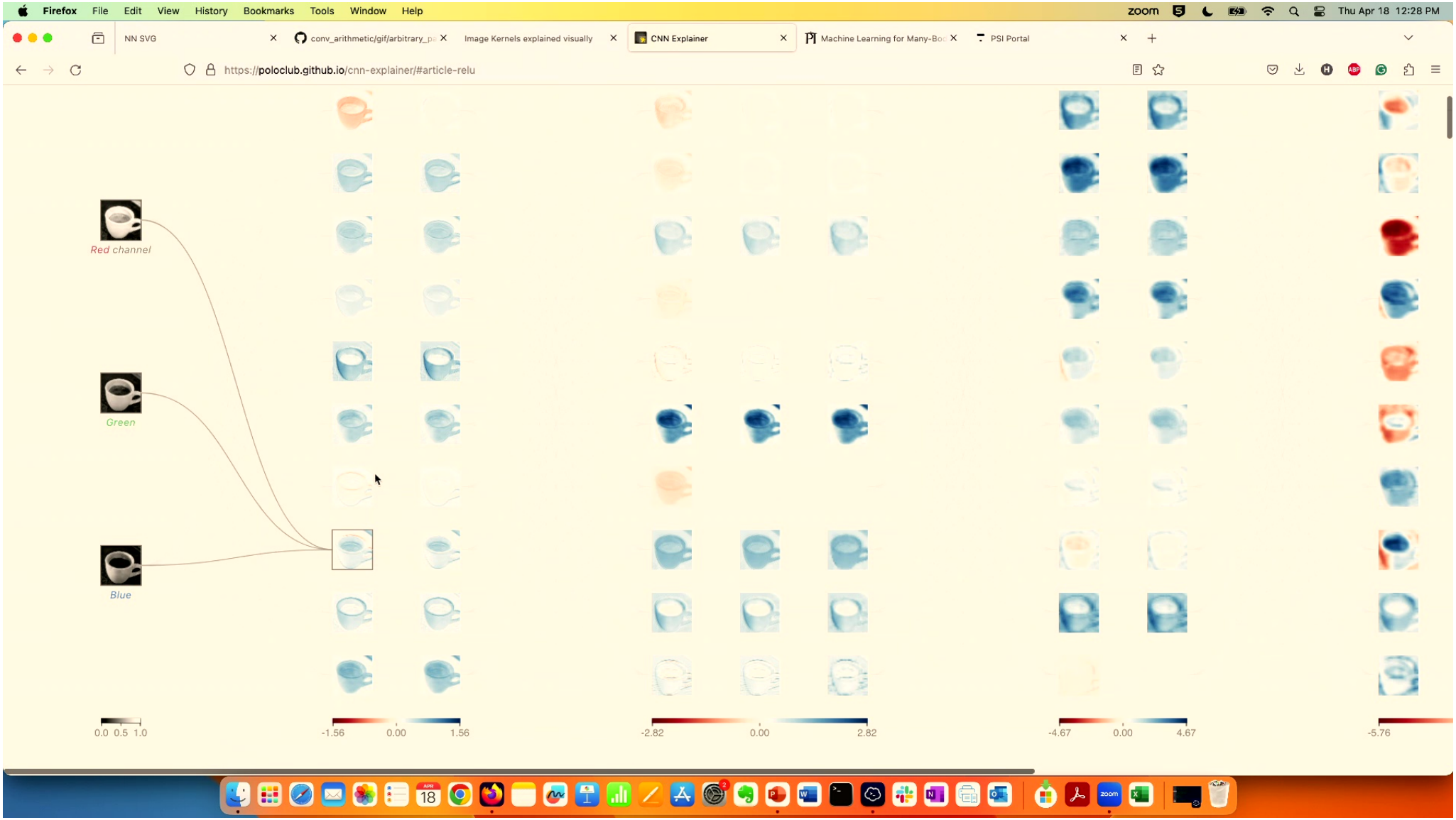

Δ: Sublattice 1
□: Sublattice 2
$N_c = 2$
(x, y)
channel (Sublattice index)
$a_{x,y,c}$
Spatial information

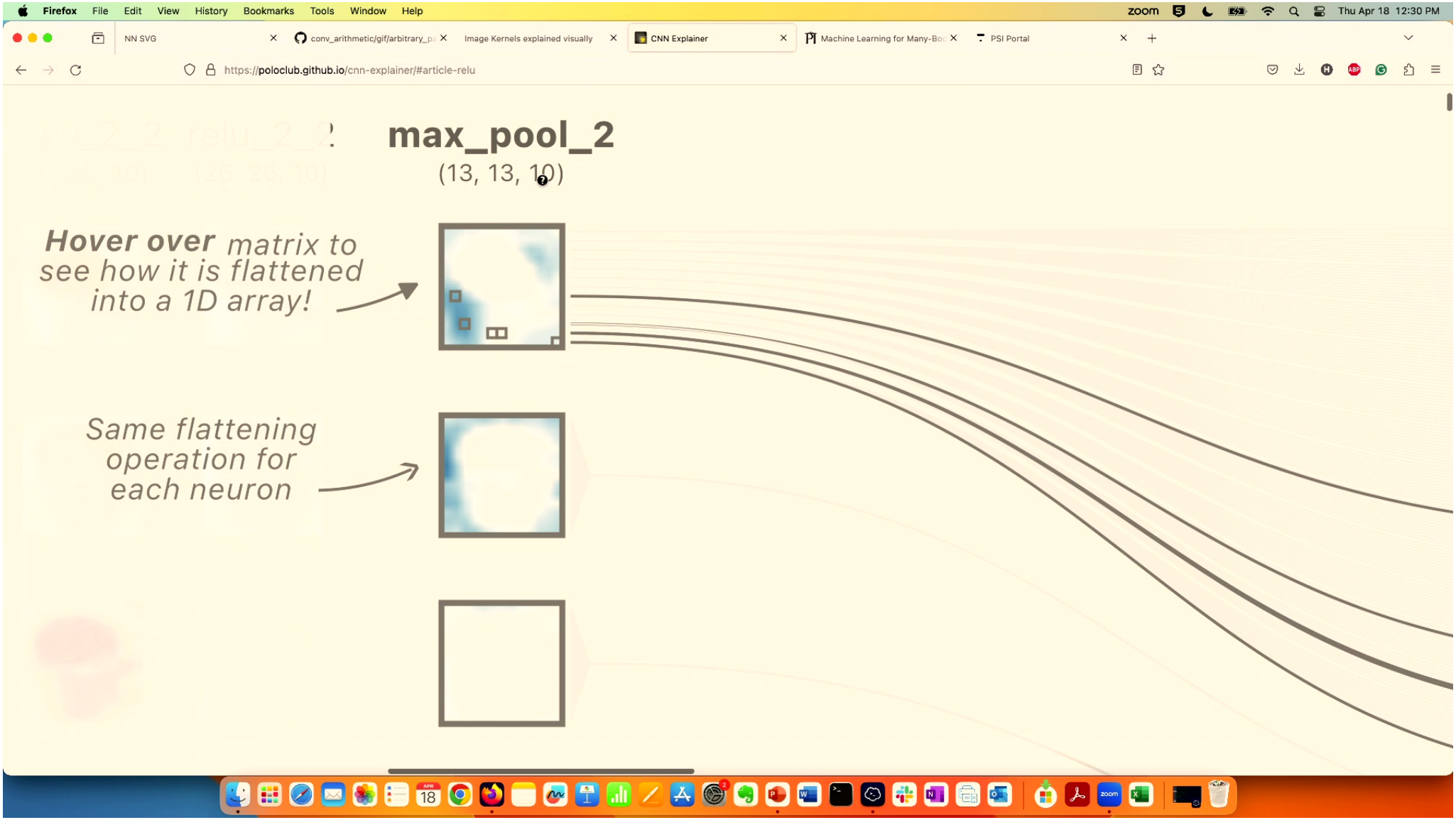

Activation function.
x index of the filter
y-index of the filter
channel or sublattice index.
Label for different filters N_g

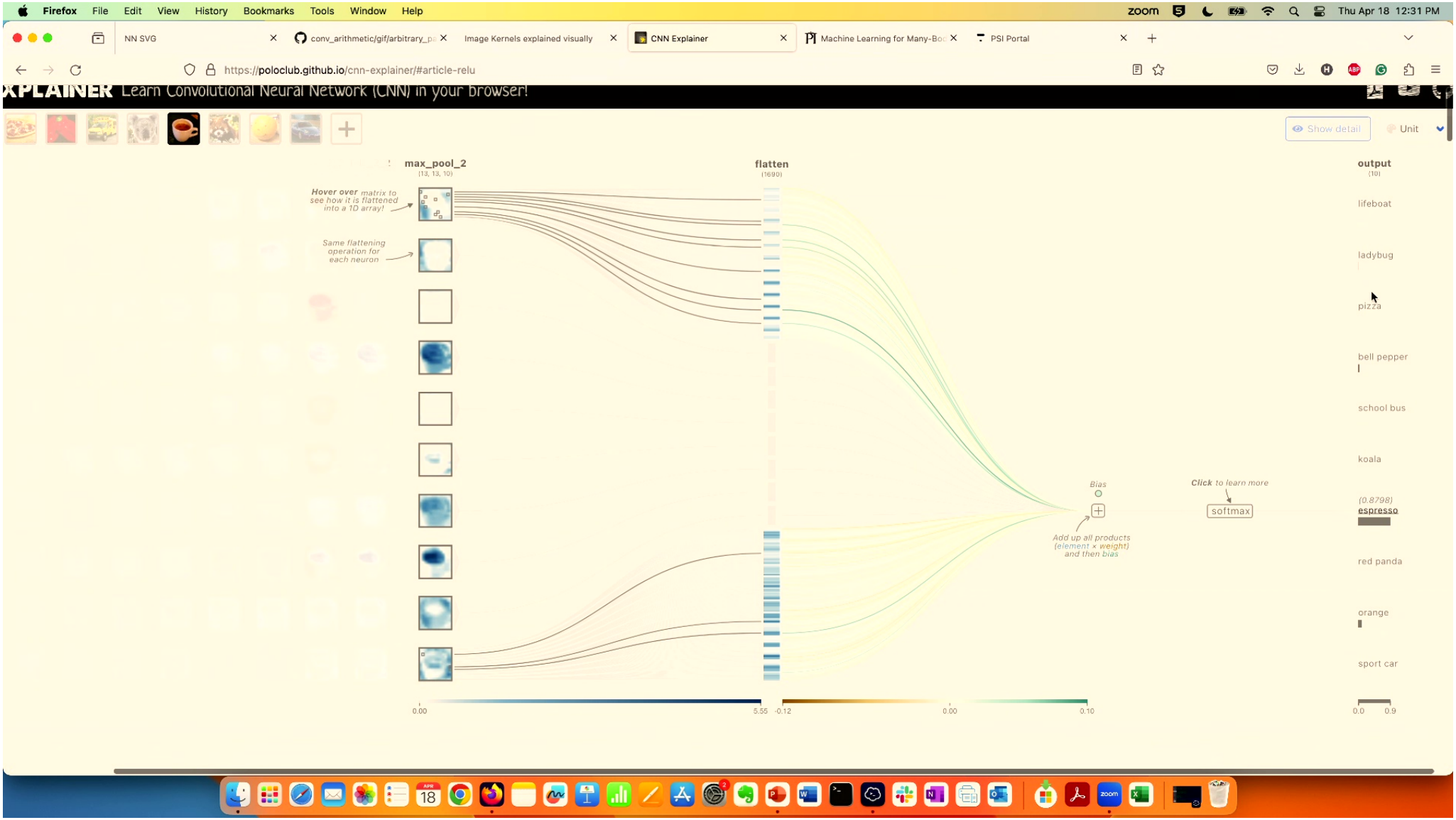

$a^{(\ell)}_{x,y,k} = g\left(\sum_{m=1}^{patch}\sum_{n=1}^{patch}\sum_{c=1}^{N_c} a^{(\ell-1)}_{x+m-1,\, y+n-1,\, c}\, W^{(\ell)}_{mnck} + b^{(\ell)}_{k}\right)$
Activation function.
x index of the filter
y-index of the filter
Channel or sublattice index.
Label for different filters N_f


No dependence on x and y (Translation Invariance)
Activation function.
x index of the filter
y-index of the filter
Channel or sublattice index.
Label for different filters N_f


Pooling
Pooling
$\square = \max(a_1, a_2, a_3, a_4), \frac{a_1 + a_2 + a_3 + a_4}{4}$

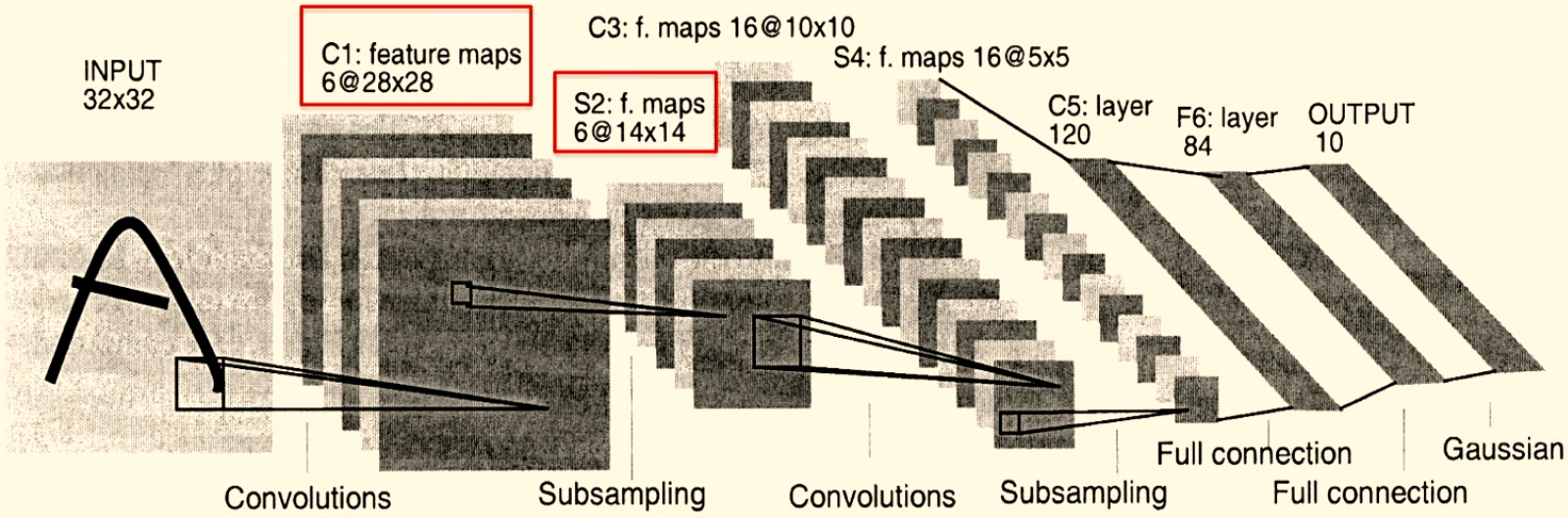

Firefox File Edit View History Bookmarks Tools Window Help
Thu Apr 18 12:28 PM
NN SVG
conv_arithmetic/gif/arbitrary_p
Image Kernels explained visually
CNN Explainer
Machine Learning for Many-Bo
PSI Portal
https://poloclub.github.io/cnn-explainer/#article-relu
Red channel
Green
Blue
0.0 0.5 1.0
-1.56 0.00 1.56
-2.82 0.00 2.82
-4.67 0.00 4.67
-5.76


Firefox File Edit View History Bookmarks Tools Window Help
Thu Apr 18 12:30 PM
NN SVG
conv_arithmetic/gif/arbitrary_p
Image Kernels explained visually
CNN Explainer
Machine Learning for Many-Bo
PSI Portal
https://poloclub.github.io/cnn-explainer/#article-relu
max_pool_2
(13, 13, 10)
Hover over matrix to see how it is flattened into a 1D array!
Same flattening operation for each neuron

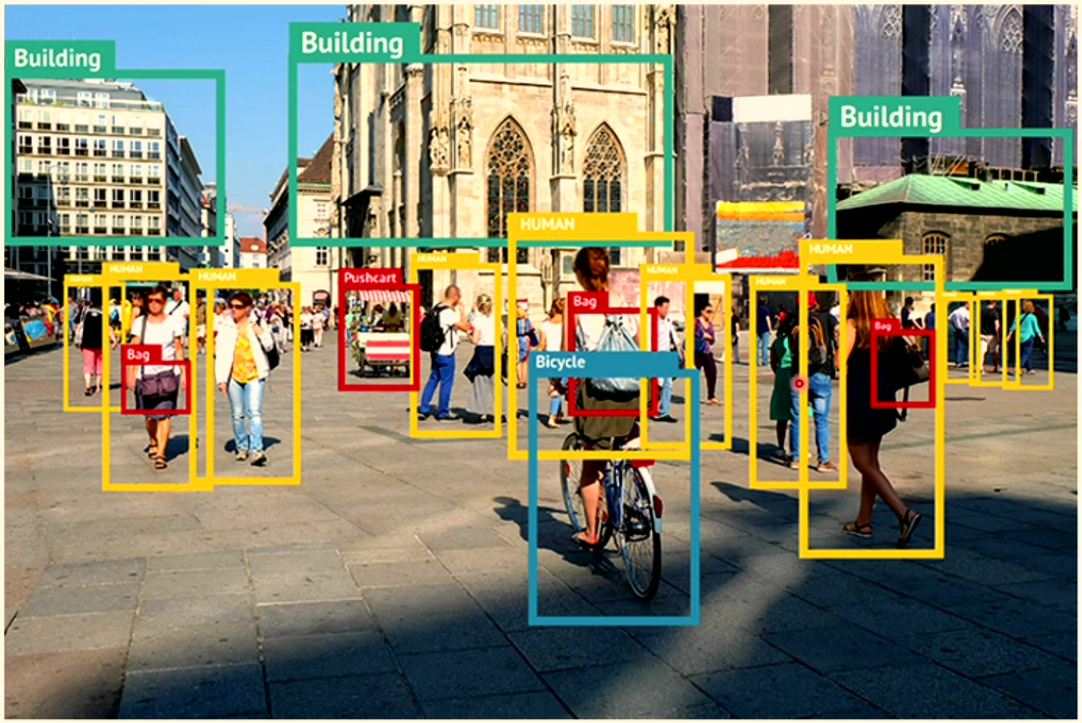

Firefox File Edit View History Bookmarks Tools Window Help
Thu Apr 18 12:31 PM
NN SVG
conv_arithmetic/gif/arbitrary_p
Image Kernels explained visually
CNN Explainer
Machine Learning for Many-Bo
PSI Portal
https://poloclub.github.io/cnn-explainer/#article-relu
XPLAINER Learn Convolutional Neural Network (CNN) in your browser!
Show detail
Unit
max_pool_2
(13, 13, 10)
flatten
(1690)
output
(10)
Hover over matrix to see how it is flattened into a 1D array!
Same flattening operation for each neuron
Bias
Add up all products (element × weight) and then bias
Click to learn more
softmax
lifeboat
ladybug
pizza
bell pepper
school bus
koala
(0.8798)
espresso
red panda
orange
sport car
0.00
5.55
-0.12
0.00
0.10
0.0
0.9

LeNet

- Here's the LeNet architecture, which was applied to handwritten digit recognition on MNIST in 1998:



Applications of CNNs



Credit: www.datasciencecentral.com

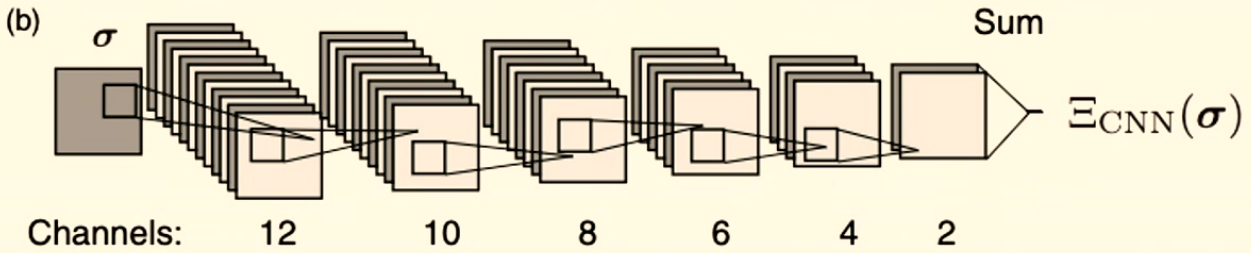
Applications of CNNs

Two-dimensional frustrated J_1-J_2 model studied with neural network quantum states

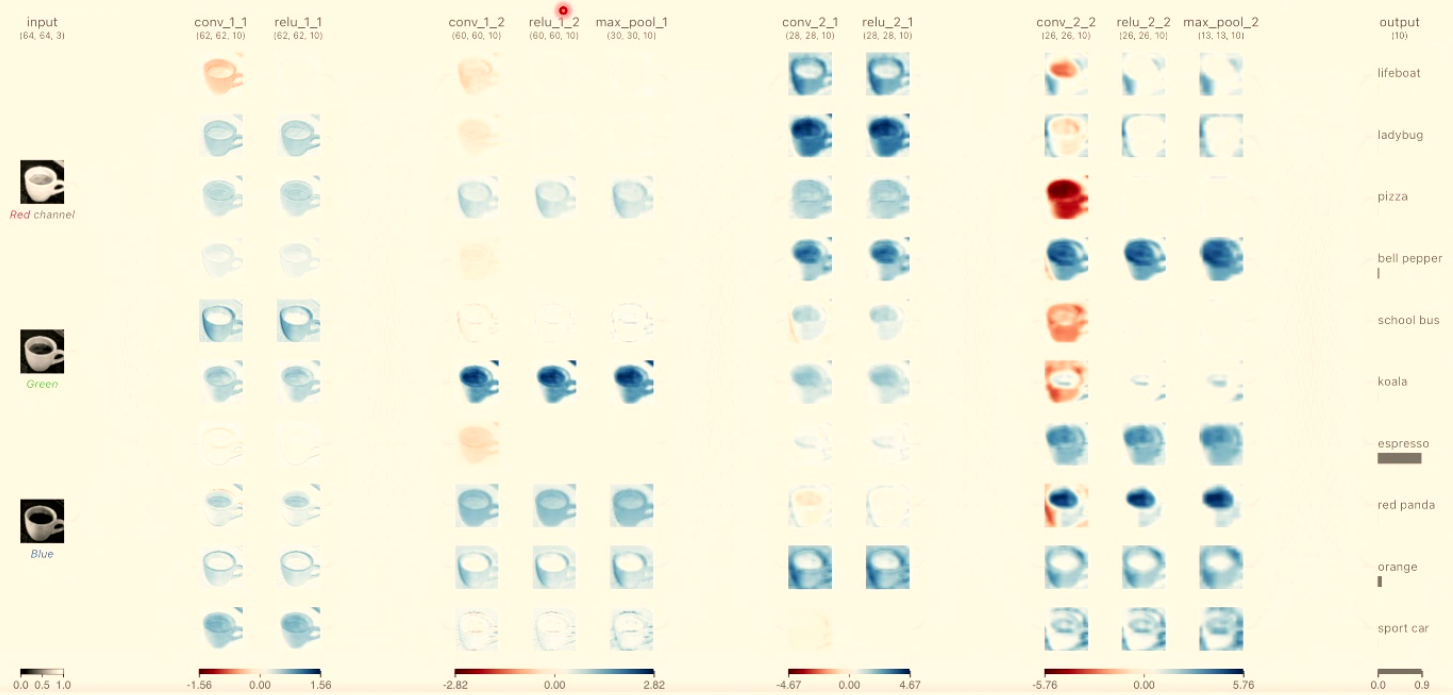
Kenny Choo,[1] Titus Neupert,[1] and Giuseppe Carleo[2]
[1]*Department of Physics, University of Zurich, Winterthurerstrasse 190, 8057 Zurich, Switzerland*
[2]*Center for Computational Quantum Physics, Flatiron Institute, 162 5th Avenue, New York, New York 10010, USA*

(Received 23 March 2019; revised manuscript received 17 July 2019; published 11 September 2019)

The use of artificial neural networks to represent quantum wave functions has recently attracted interest as a way to solve complex many-body problems. The potential of these variational parametrizations has been supported by analytical and numerical evidence in controlled benchmarks. While approaching the end of the early research phase in this field, it becomes increasingly important to show how neural-network states perform for models and physical problems that constitute a clear open challenge for other many-body computational methods. In this paper, we start addressing this aspect, concentrating on a presently unsolved model describing two-dimensional frustrated magnets. Using a fully convolutional neural network model as a variational ansätz, we study the frustrated spin-1/2 J_1-J_2 Heisenberg model on the square lattice. We demonstrate that the resulting predictions for both ground-state energies and properties are competitive with, and often improve upon, existing state-of-the-art methods. In a relatively small region in the parameter space, corresponding to the maximally frustrated regime, our ansätz exhibits comparatively good but not the best performance. The gap between the complexity of the models adopted here and those routinely adopted in deep-learning applications is, however, still substantial, such that further improvements in future generations of neural-network quantum states are likely to be expected.



Demonstration of CNNs



https://poloclub.github.io/cnn-explainer/