CONICET

ML Fundamentals

Dra María Graciela Molina
FACET-UNT / CONICET
Tucumán Space Weather Center - TSWC

https://spaceweather.facet.unt.edu.ar/
IG -> @spaceweatherargentina

gmolina@herrera.unt.edu.ar

Outline

- Deep Learning
- Recurrent NN
- LSTM
- Convolutional NN

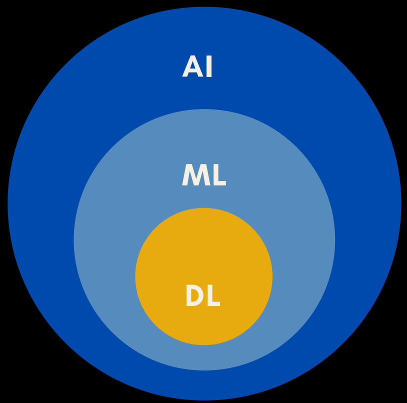

María Graciela Molina
gmolina@herrera.unt.edu.ar


Jorge H. Namour
jnamour@herrera.unt.edu.ar

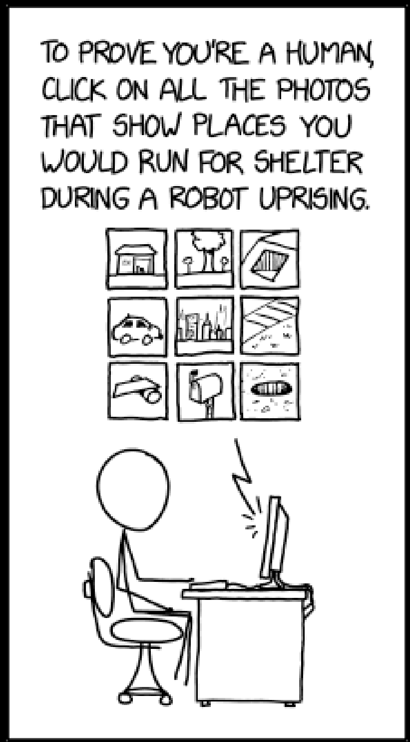

Cecilia Jarne
cecilia.jarne@unq.edu.ar


Ticiano Torres Peralta
ttorres@herrera.unt.edu.ar

Deep Learning



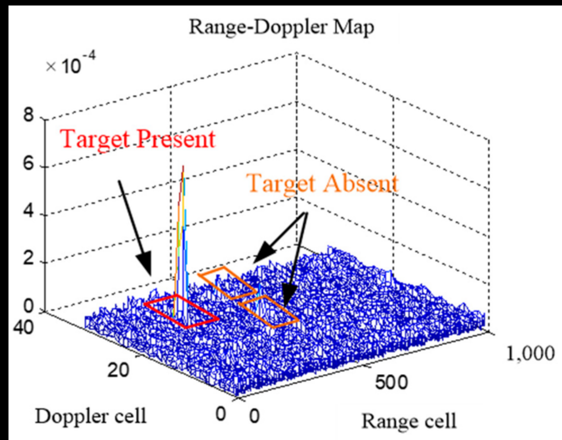
Extract patterns from data using neural networks

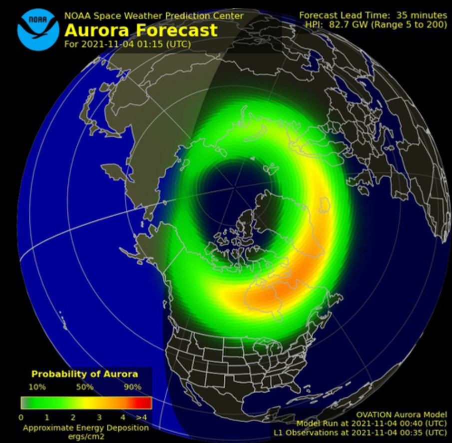


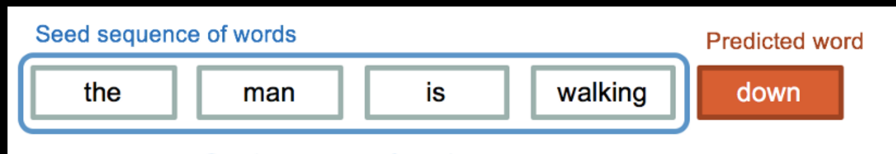
- https://this-person-does-not-exist.com/en

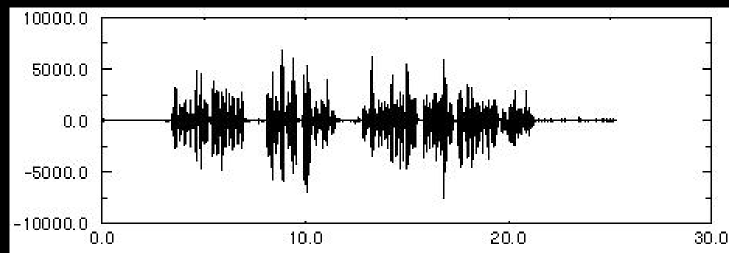
Recurrent Neural Networks

- Processing a sequence of data x(t)= x(1), . . . , x(τ)
- Recurrent -> perform the same task for every element of a sequence, with the output being depended on the previous computations.
- Have a “memory”









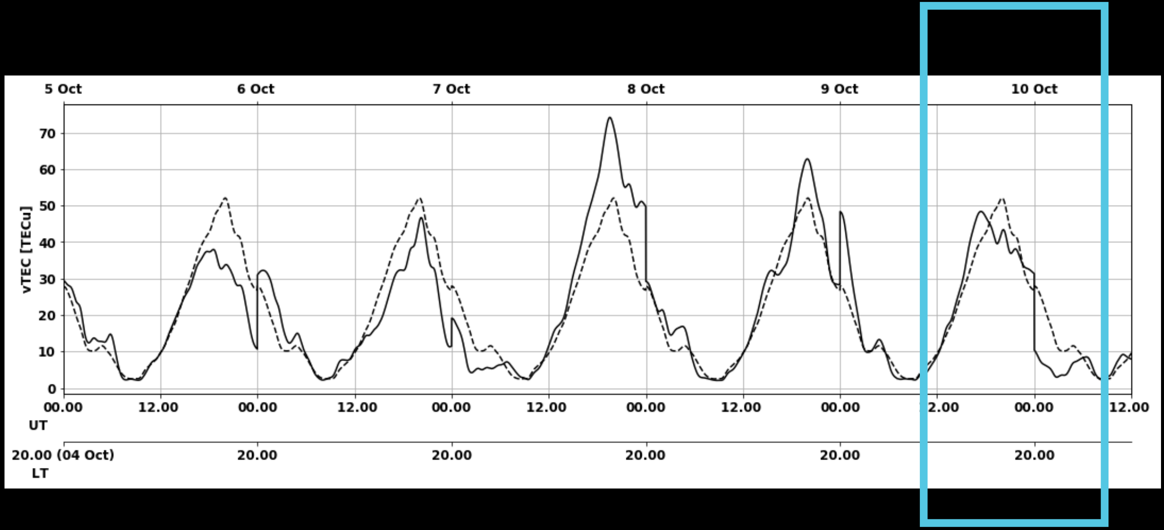
Recurrent Neural Networks

RNNs as an approach to sequence modeling problems

We want to forecast this

To model sequences, we need to:

- Handle variable-lenght sequences
- Track long-term dependences
- Mantain information about order
- Share parameters across the sequence



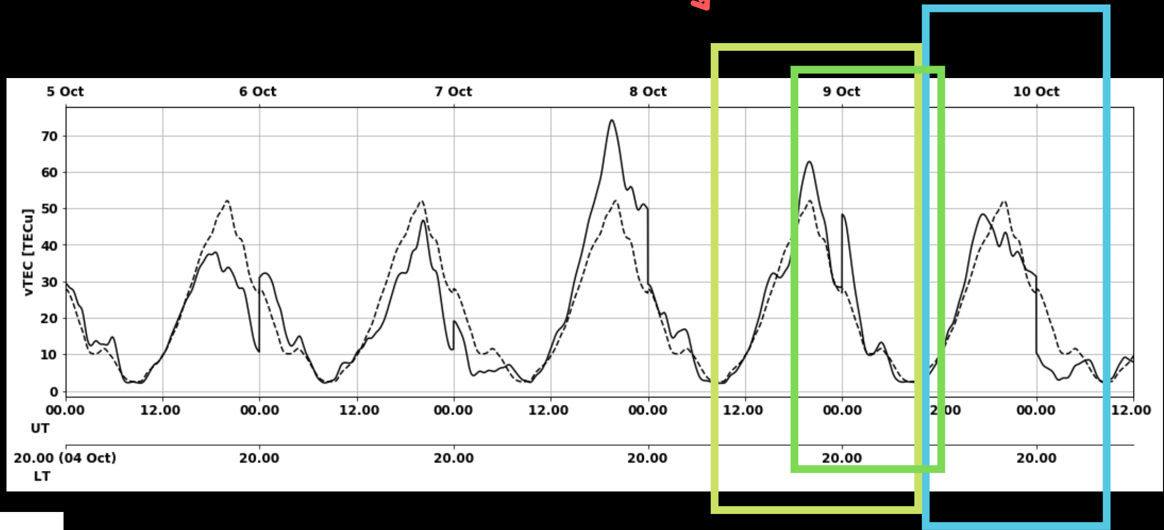
Recurrent Neural Networks

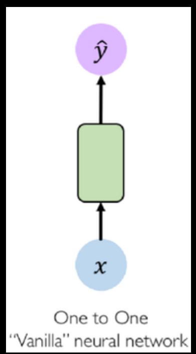
RNNs as an approach to sequence modeling problems

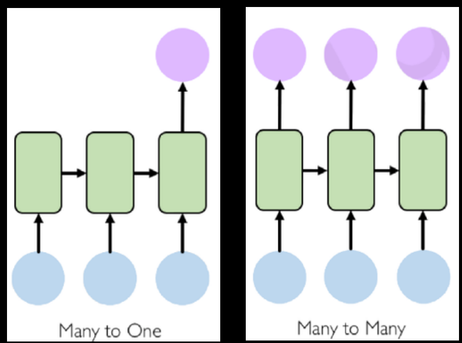
how many steps? (lenght)

To model sequences, we need to:

- Handle **variable-lenght** sequences
- Track **long-term** dependences
- Mantain information about **order**
- **Share parameters** across the sequence







and more arquitecutes

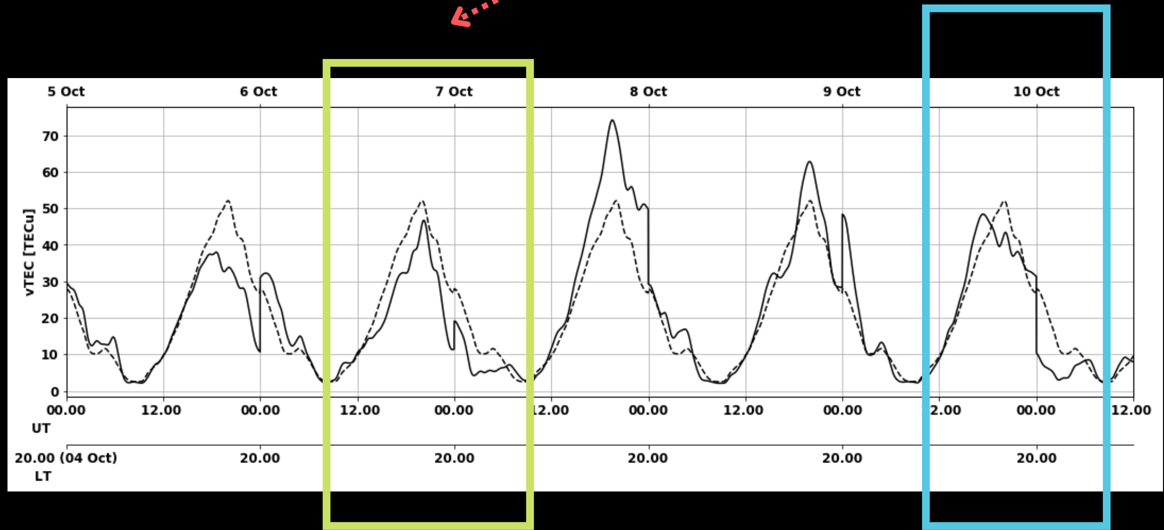
Recurrent Neural Networks

RNNs as an approach to sequence modeling problems

how important is the information on the (far) past ?

To model sequences, we need to:

- Handle variable-lenght sequences
- Track long-term dependences
- Mantain information about order
- Share parameters across the sequence



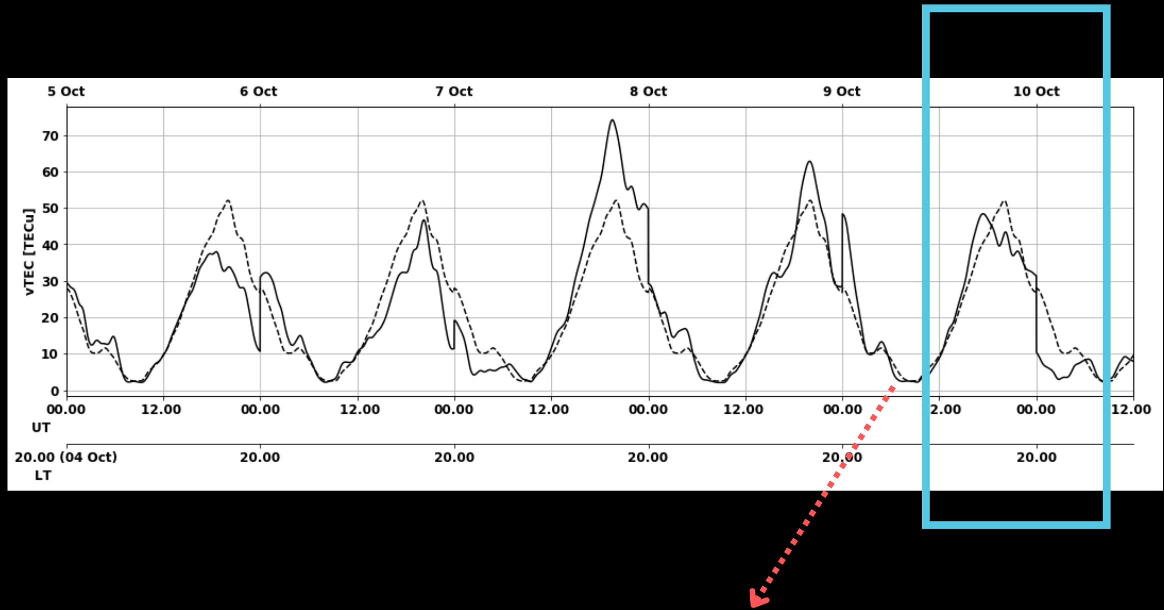
E.g. Ionosphere regular behaviour (daily, seasonal, solar cycle, ...)

Recurrent Neural Networks

RNNs as an approach to sequence modeling problems

To model sequences, we need to:

- Handle variable-lenght sequences
- Track long-term dependences
- Mantain information about order
- Share parameters across the sequence



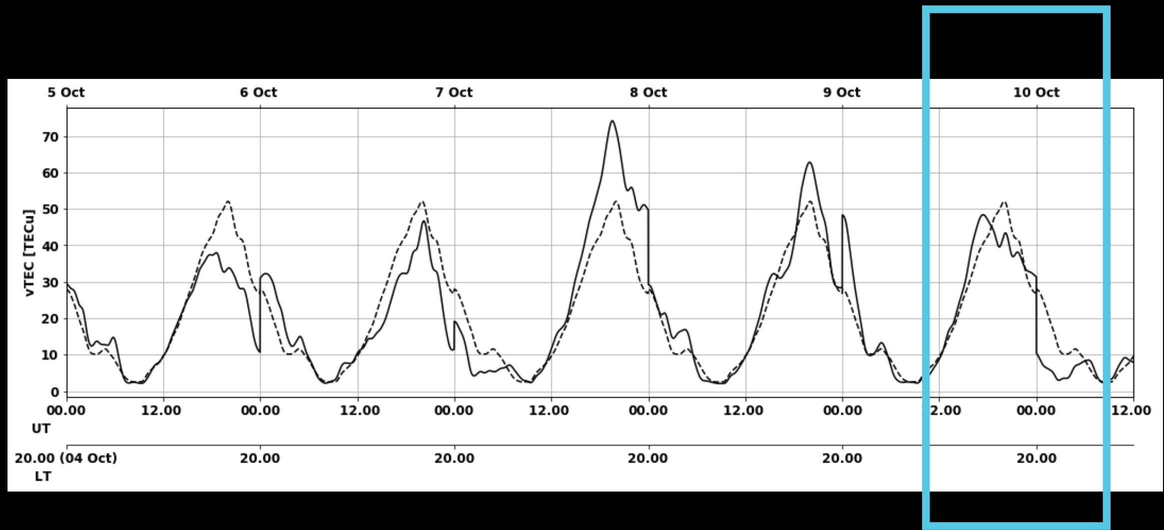
vTEC (t-2), vTEC (t-1), vTEC (t-0)   <> vTEC (t-0), vTEC (t-2), vTEC (t-1)

Recurrent Neural Networks

RNNs as an approach to sequence modeling problems

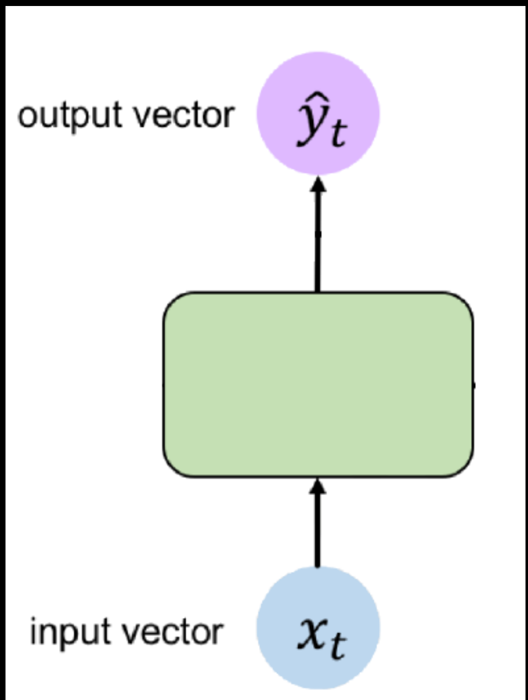
To model sequences, we need to:

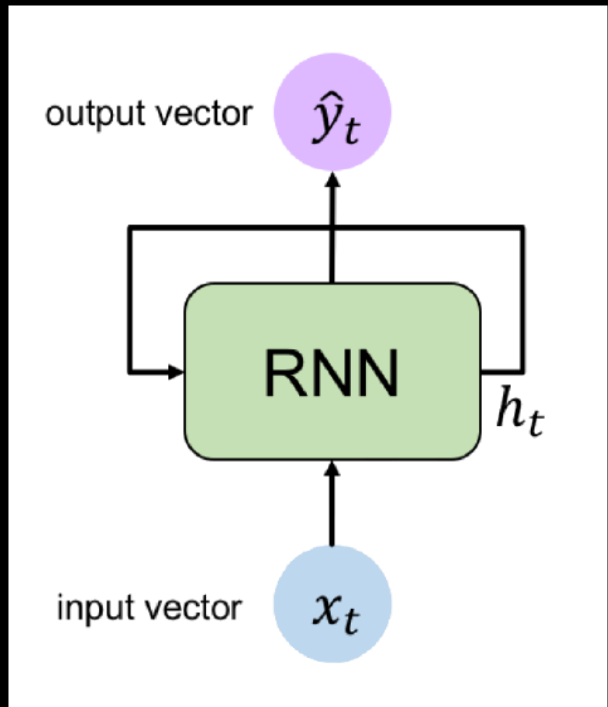
- Handle variable-lenght sequences
- Track long-term dependences
- Mantain information about order
- Share parameters across the sequence



RNNs have a state (h_t), that is updated at each time as a sequence is processed using the same parameters each time step

Recurrent Neural Networks

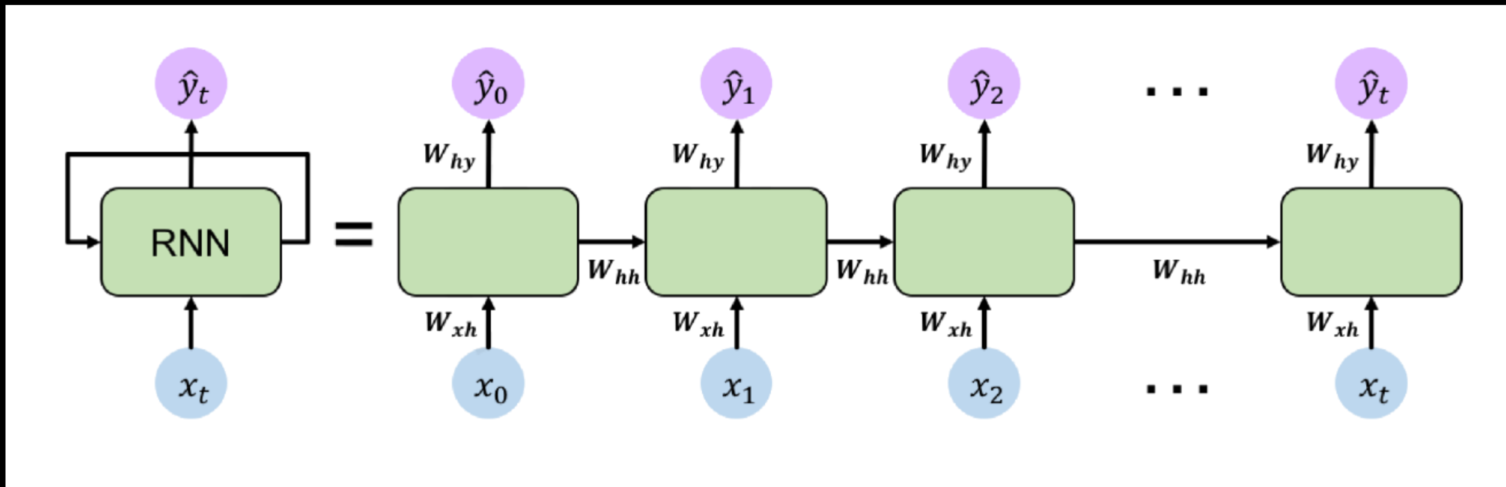




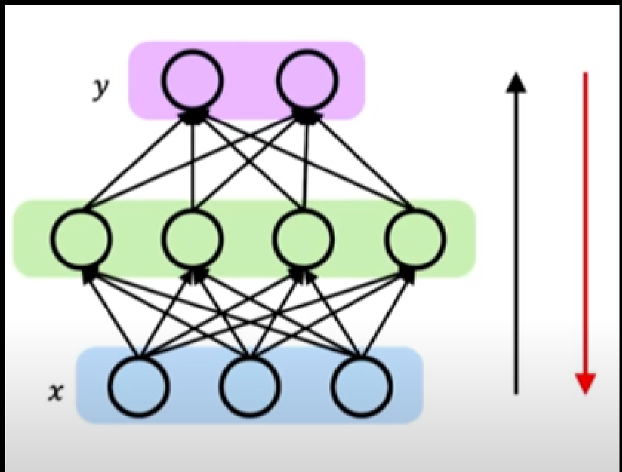
$$\underbrace{\boxed{h_t}}_{\text{cell state}} = \underbrace{\boxed{f_W}}_{\text{function parameterized by W}}(\underbrace{\boxed{h_{t-1}}}_{\text{old state}}, \underbrace{\boxed{x_t}}_{\text{input vector at time step } t})$$

- Apply a recurrent relation at every time step to process a sequence

Recurrent Neural Networks



Backpropagation through time: long time dependences



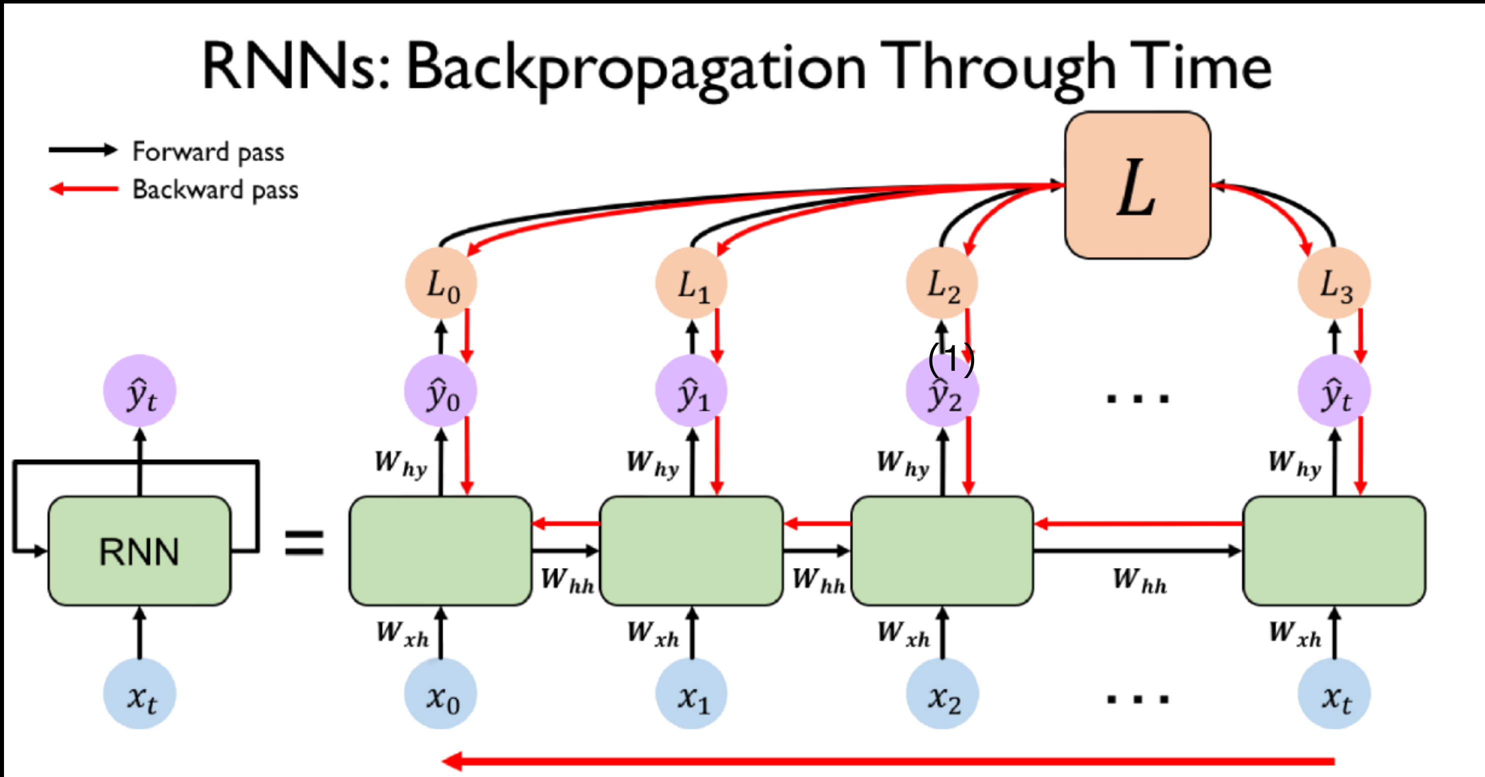
$$\boxed{\begin{gathered} W^* = \underset{W}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\left(f\left(x^{(i)}; W\right), y^{(i)}\right) \\ W^* = \underset{W}{\operatorname{argmin}}\, J(W) \end{gathered}}$$

Algorithm

1. Initialize weights randomly $\sim\mathcal{N}(0, \sigma^2)$
2. Loop until convergence:
3. Compute gradient, $\frac{\partial J(W)}{\partial W}$
4. Update weights, $W \leftarrow W - \eta \frac{\partial J(W)}{\partial W}$
5. Return weights

- take the gradient of loss with respect to each parameter
- shift parameters in order to minimize loss

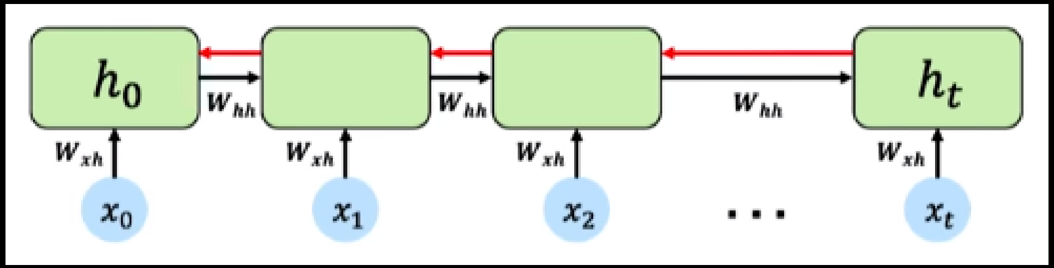
Backpropagation through time: long time dependences



- Compute individual l_i for individual time steps and sum them
- Backpropagate errors individually for each time step and then to all the time steps to the beginning of the sequence.

https : //kharshit.github.io/blog /2019/02/22/backpropagation – through – time

Backpropagation through time: long time dependences



- high computation time!

- Many values >>1 -> exploding gradient (*)
- Many values <<1 -> vanishing gradient

(*) Gradient clipping is a simple technique: If the gradient gets too large, we rescale it to keep it small.

Vanishing gradient problem

Multiply many small numbers together

Errors due to further back time steps have smaller and smaller gradientes

Bias parameters to capture short -term dependencies

How to tackle the problem:

- Activation function
- Weight initialization
- Network architecture

Vanishing gradient problem

TSWC, 2022

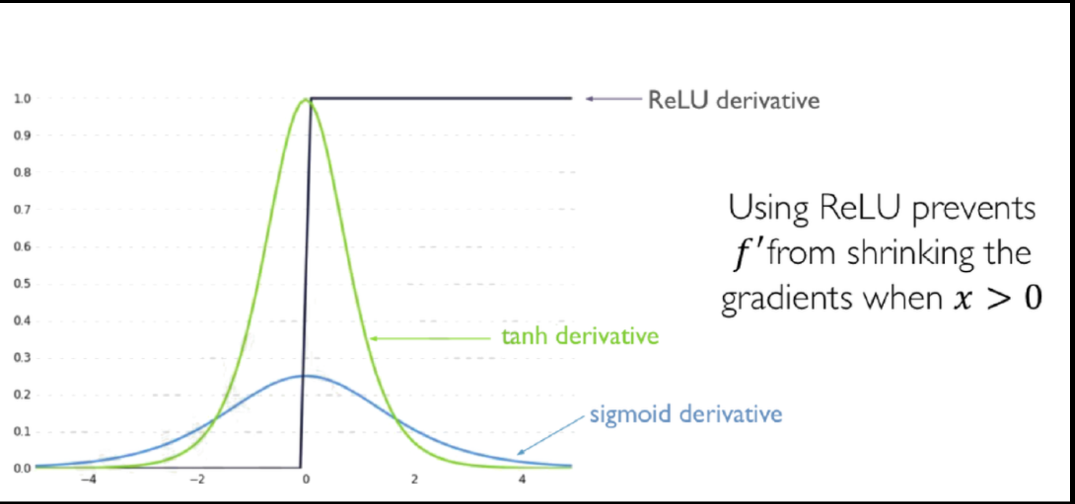
Multiply many small numbers together

Errors due to further back time steps have smaller and smaller gradientes

Bias parameters to capture short -term dependencies

How to tackle the problem:

- Activation function
- Weight initialization
- Network architecture



TSWC, 2022

Vanishing gradient problem

Multiply many small numbers together

Errors due to further back time steps
have smaller and smaller gradientes

Bias parameters to capture short -term dependencies (we "loose" long-term dependencies)

How to tackle the problem:

- Activation function
- Weight initialization
- Network architecture

$$I_n = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

Initialize weigths to identity matriz
Initialize biases to zero

prevent the weights to shrinking to zero

TSWC, 2022

Vanishing gradient problem

Multiply many small numbers together

↓

Errors due to further back time steps have smaller and smaller gradientes

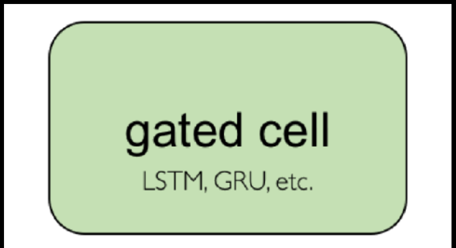
↓

Bias parameters to capture short -term dependencies (we "loose" long-term dependencies)

How to tackle the problem:

- Activation function
- Weight initialization
- Network architecture

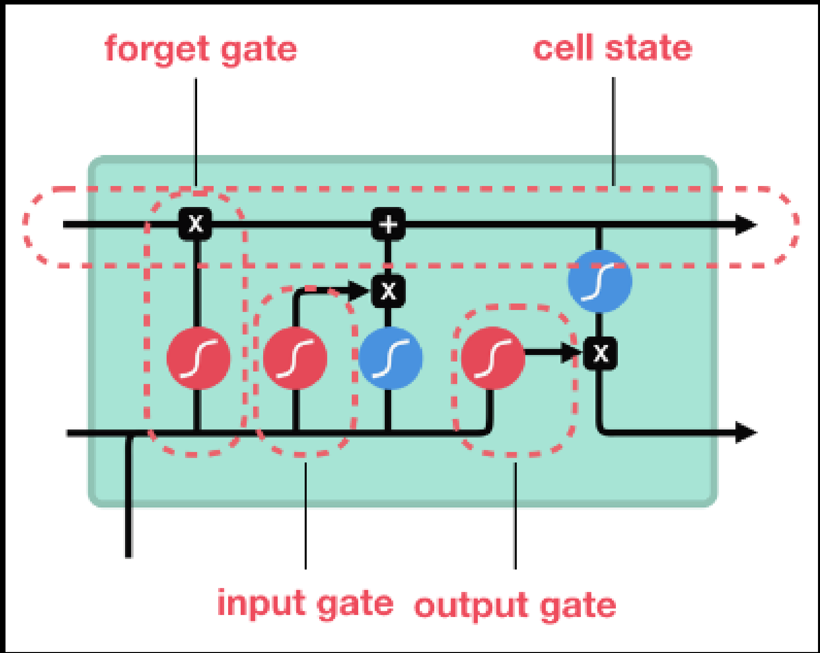
more rubust solution



- Use a more complex recurrent unit with gates to control what information is passed through.
- gates selectively add or remove information within each recurrent uniit

Long short term memory (LSTM)

Gates: 1) Forget 2) Input (store) 3)Update 4) Output



How it works:

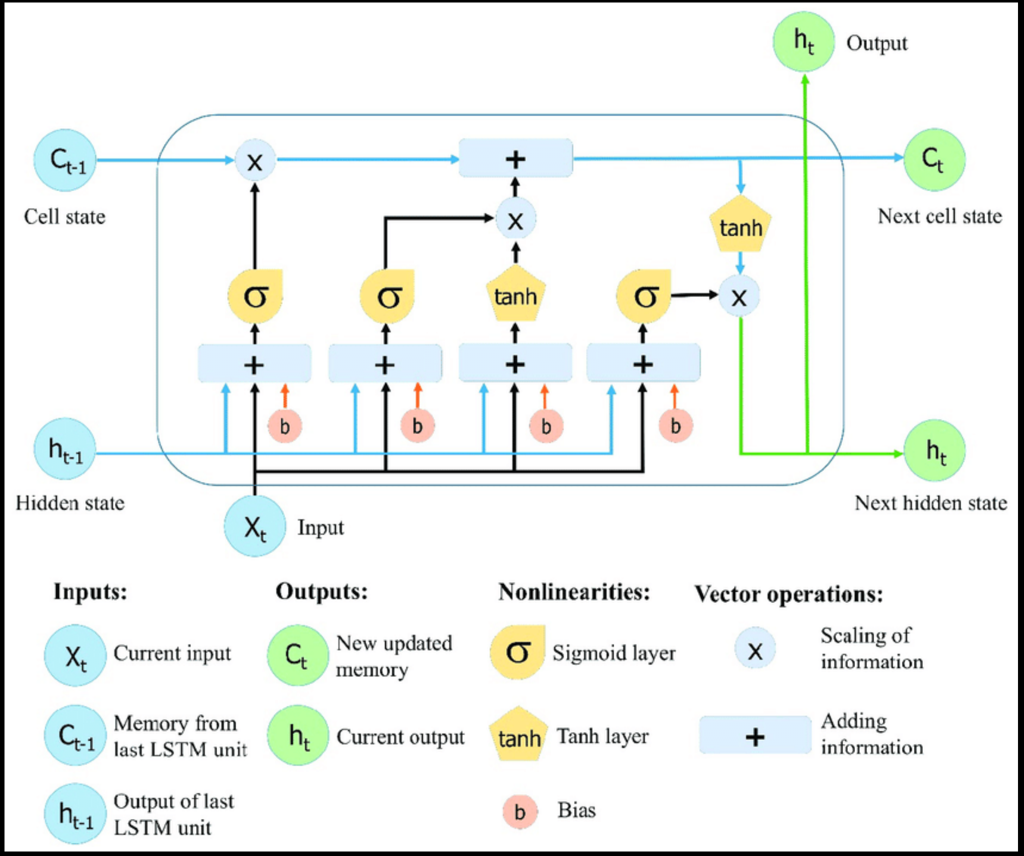
1)Maintain a cell state
2)Use gates to control the flow of information

- Forget gate gets rid of irrelevant information
- Store relevant information from the current input
- Selectively update cell state

Output gate returns a filtered version of the cell state
3) Backpropagation TT with partially uninterrupted gradient flow

Long short term memory (LSTM)

Gates: 1) Forget 2) Input (store) 3)Update 4) Output



How it works:

1)Maintain a cell state
2)Use gates to control the flow of information
- Forget gate gets rid of irrelevant information
- Store relevant information from the current input
- Selectively update cell state

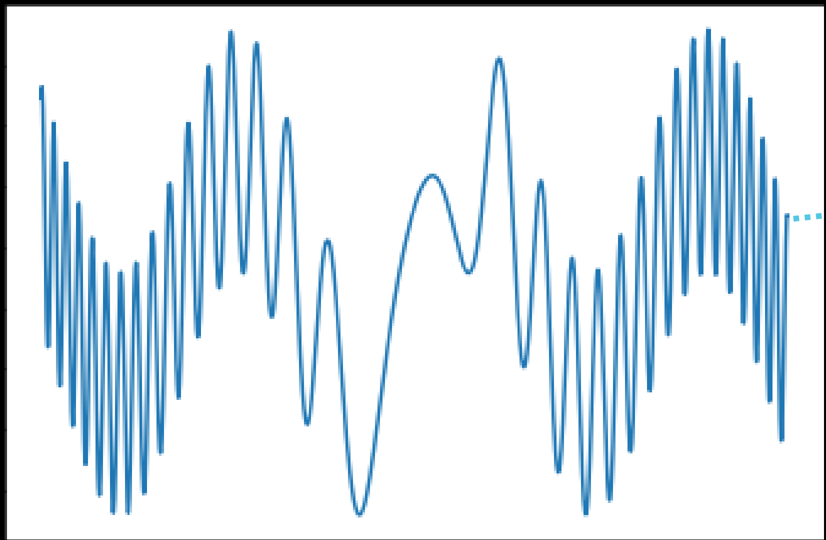
Output gate returns a filtered version of the cell state
3) Backpropagation TT with partially uninterrupted gradient flow

LSTM "not all that glitters is gold"

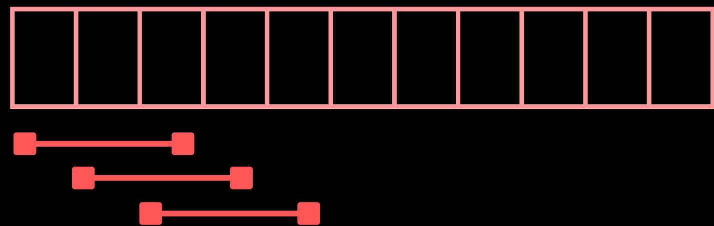
- << vanishing gradient problem .... but it doesn't completely remove it.
- >> computational resources
- affected by different random weight initialization
- Drop-out difficult to implement
- Prone to overfitting
- << performance for very long time problems

LSTM example

vanilla LSTM



training set t in [-10, 10)
test set t=10

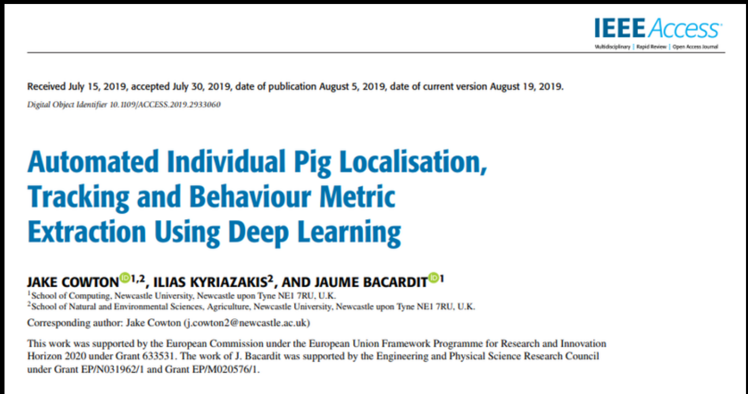


the training set is an array of
sequences (len = 3)

Convolutional Neural Networks

Computer Vision. Some applications:

- Facial detection and recognition
- Healthcare, medicine and biology
- Self-driving vehicles



IEEE Access

Multidisciplinary | Rapid Review | Open Access Journal

Received July 15, 2019, accepted July 30, 2019, date of publication August 5, 2019, date of current version August 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2933060

Automated Individual Pig Localisation, Tracking and Behaviour Metric Extraction Using Deep Learning

JAKE COWTON[1,2], ILIAS KYRIAZAKIS[2], AND JAUME BACARDIT[1]

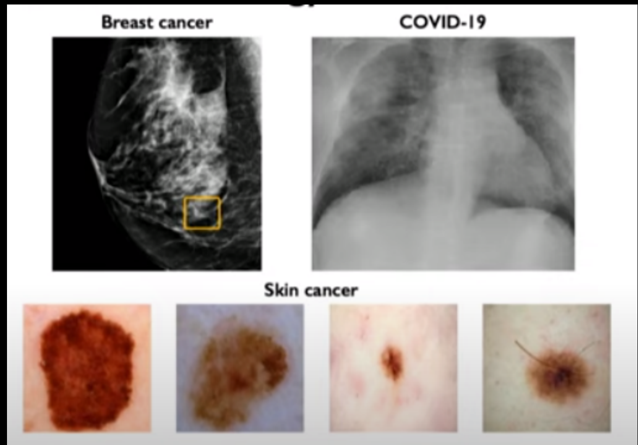
[1]School of Computing, Newcastle University, Newcastle upon Tyne NE1 7RU, U.K.

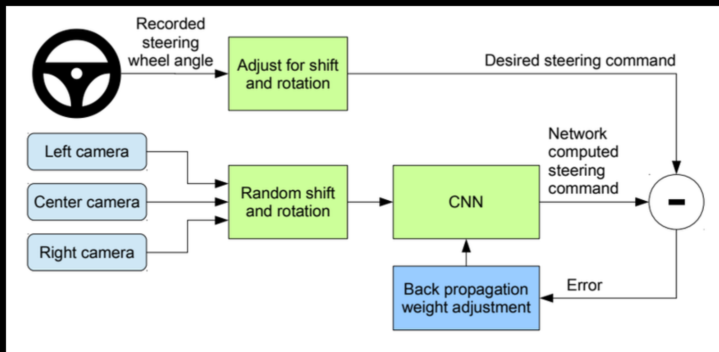
[2]School of Natural and Environmental Sciences, Agriculture, Newcastle University, Newcastle upon Tyne NE1 7RU, U.K.

Corresponding author: Jake Cowton (j.cowton2@newcastle.ac.uk)

This work was supported by the European Commission under the European Union Framework Programme for Research and Innovation Horizon 2020 under Grant 633531. The work of J. Bacardit was supported by the Engineering and Physical Science Research Council under Grant EP/N031962/1 and Grant EP/M020576/1.

FIGURE 8. Four sample images from our pig detection test set processed by the Faster R-CNN with the feature extraction layers pre-trained on ImageNet, the rest pre-trained on Pascal Visual Object Classes Challenge 2007 and an additional fully-connected layer for the pig dataset. Detections to the left of the red wall are ignored. The top left image is from the low-light test segment. The top right image is from the densely packed test-segment. The bottom left image is from the overexposed test segment. The bottom right image is from the many pigs test segment.





End to End Learning for Self-Driving Cars

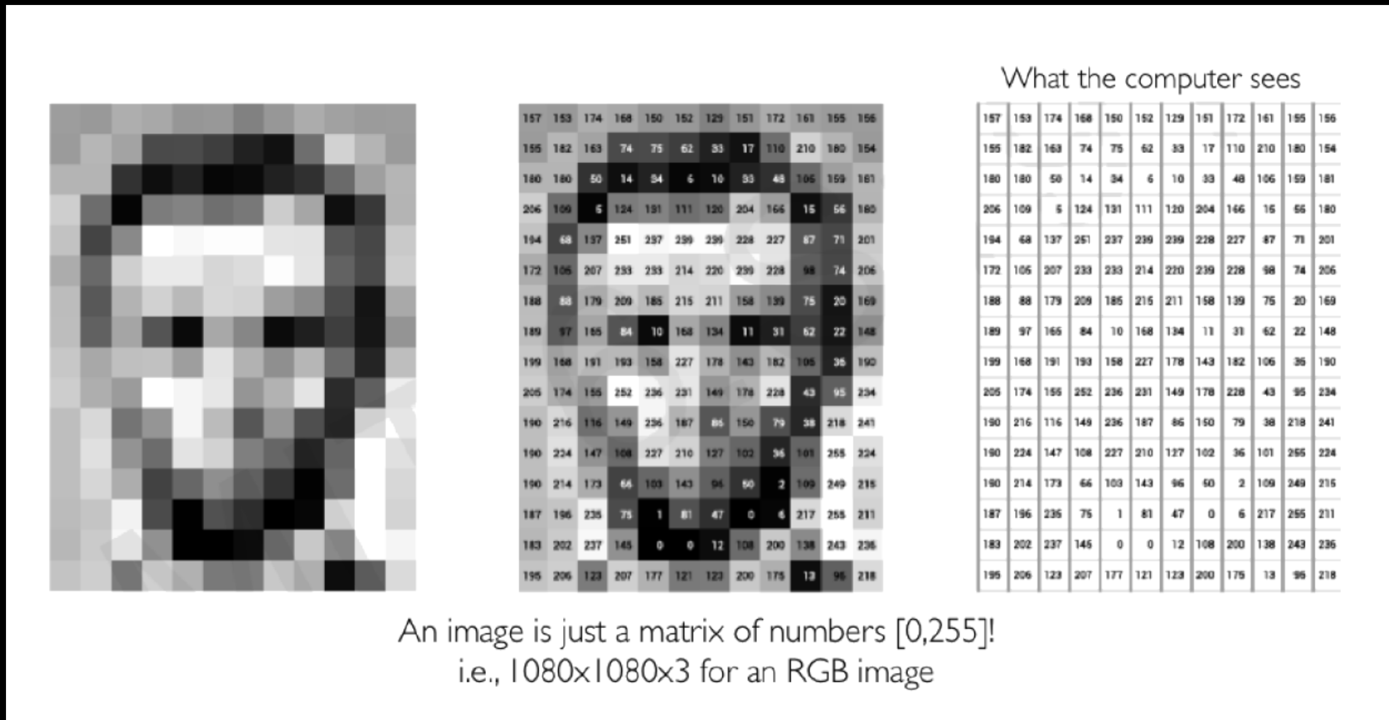
We trained a convolutional neural network (CNN) to map raw pixels from a single front-facing camera directly to...

arXiv.org

Convolutional Neural Networks

Computer Vision. Some applications:

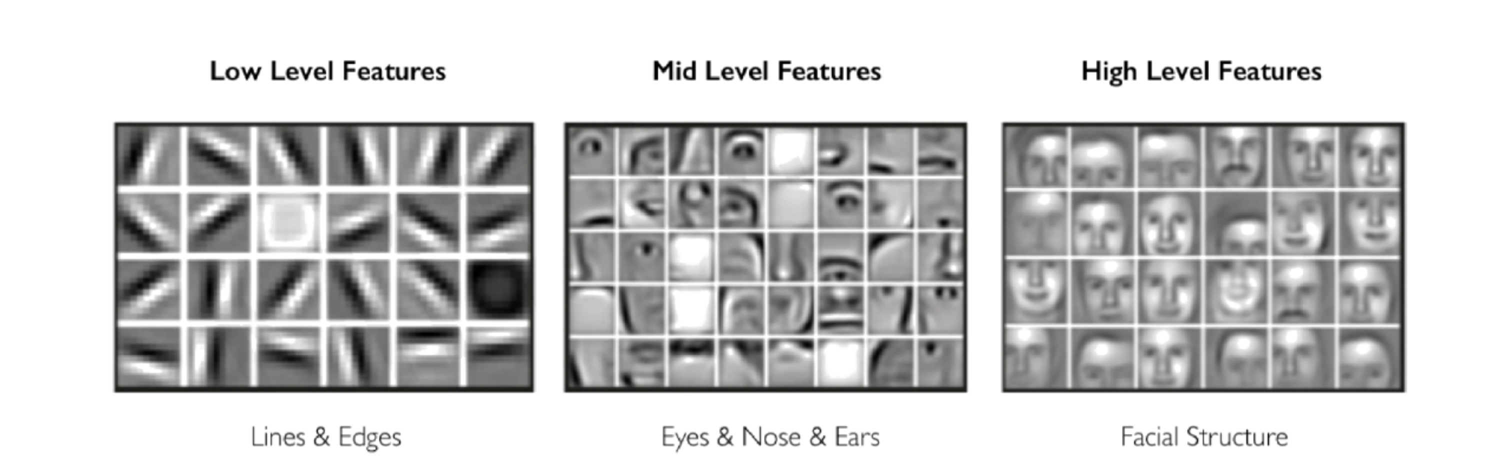
- Facial detection and recognition
- Healthcare, medicine and biology
- Self-driving vehicles



Features

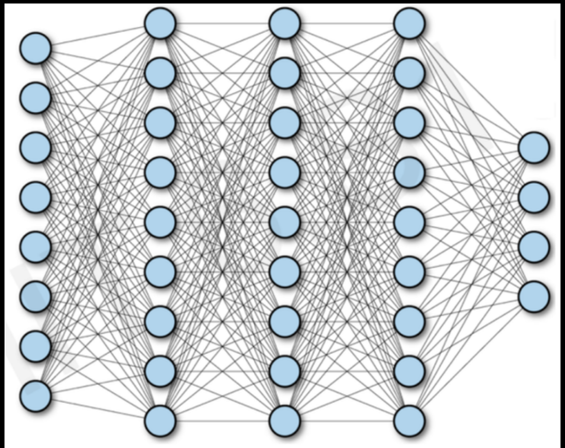
(As in other NN problems)

- Regression
- Classification
- High level feature detection



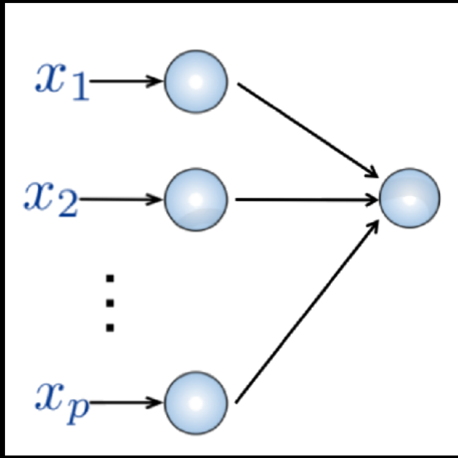
feature extraction from the data!
Learn hierarchy of features!

Fully connceted NN



Input:
- 2D image
- vector of pixel values

(flatten the image)

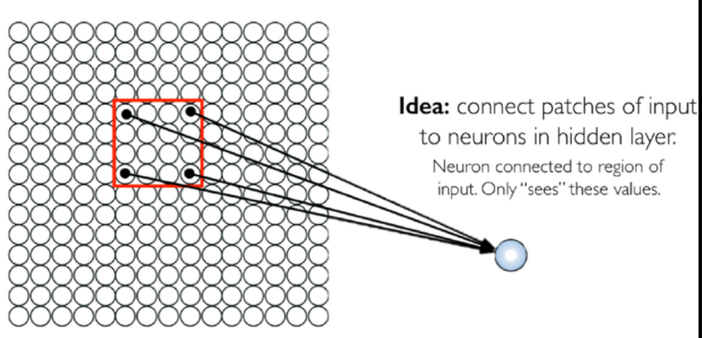


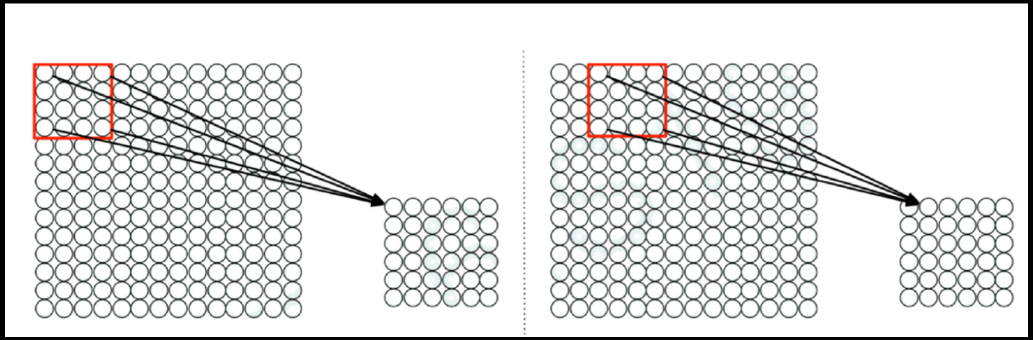
Fully connected:
- Connect neuron in hidden layer to all neurons in the input layer
- No spatial information!
- Lot of parameters

How to add spatial structure in the input?

Using spatial structure

Input:
- 2D image
- Array of pixel values

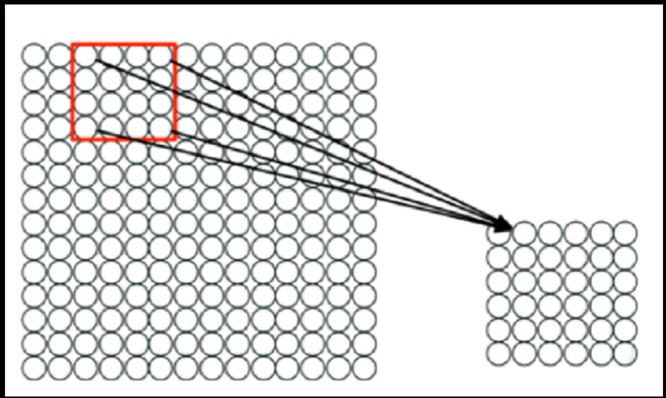




Sliding window to define conections,(connect patch in input layer to a single neuron)

The key: how can we weight the patch to detect particular features?

Using spatial structure

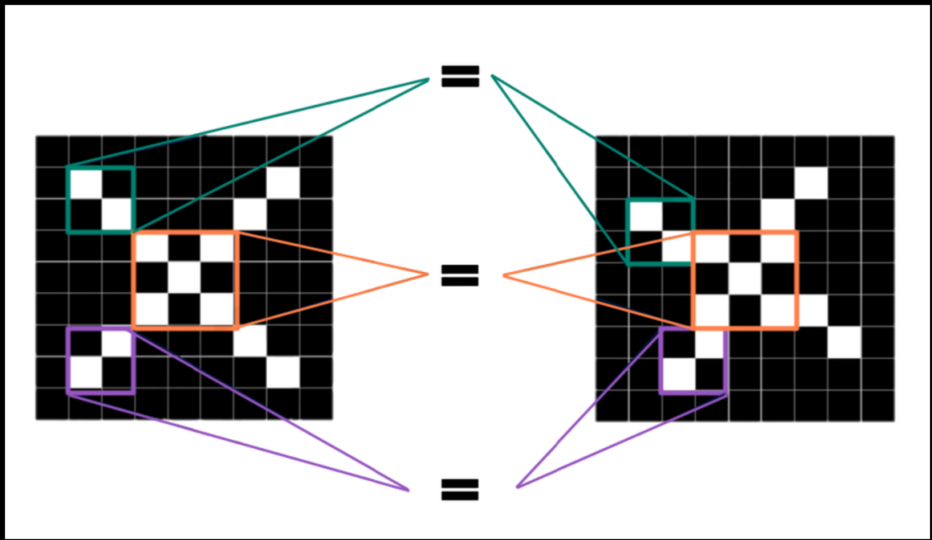


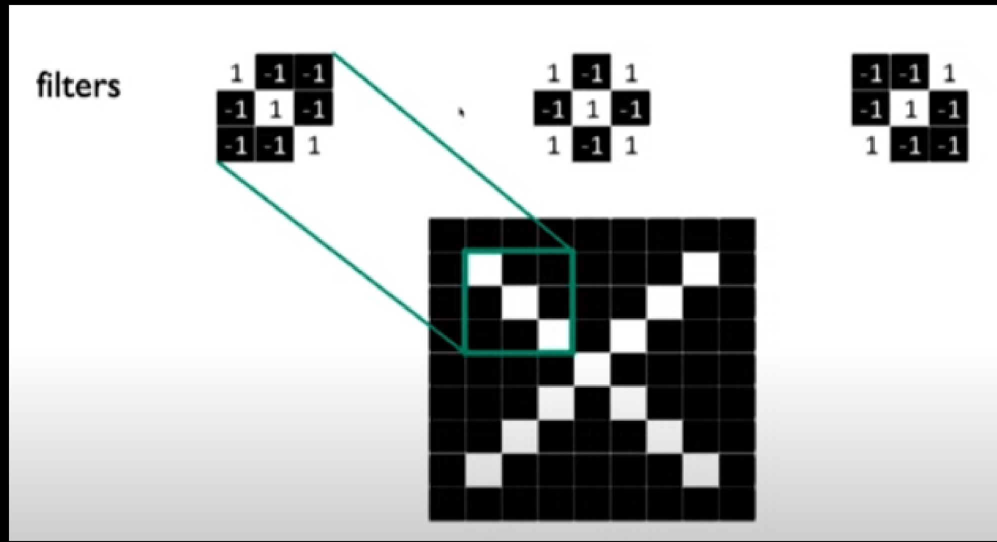
- Filter of size 4x4: 16 different weights
- Apply this same filter to a 4x4 patches in input
- Shift by 2 pixels for next patch

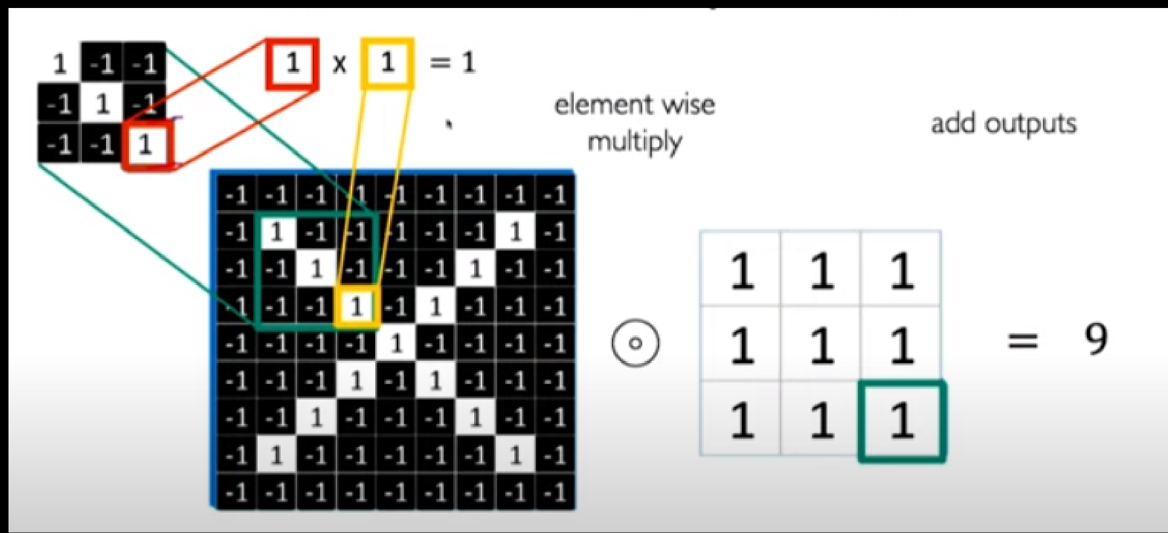
This "patchy" operation is convolution

- Apply a set of weights (a filter) to extract local features
- Use multiple filters to extract different features
- Spatially share parameters of each filter

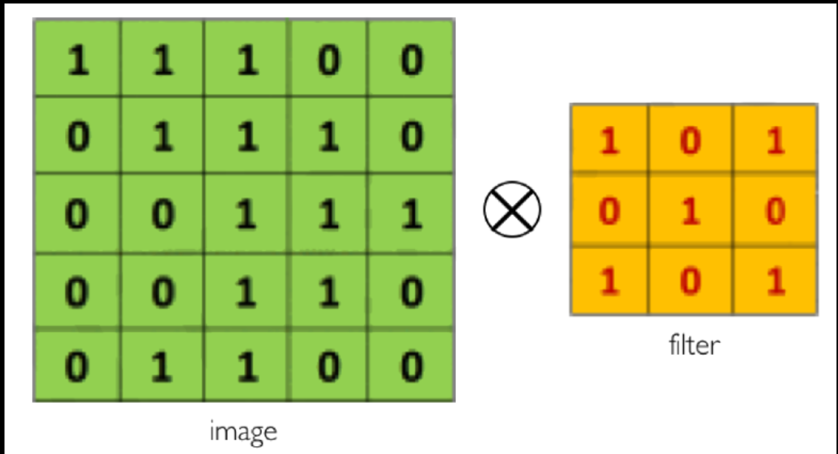
Convolutional operation

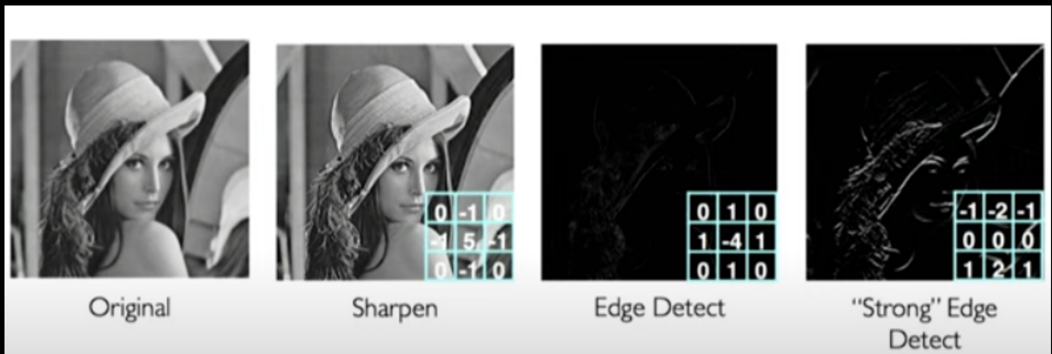


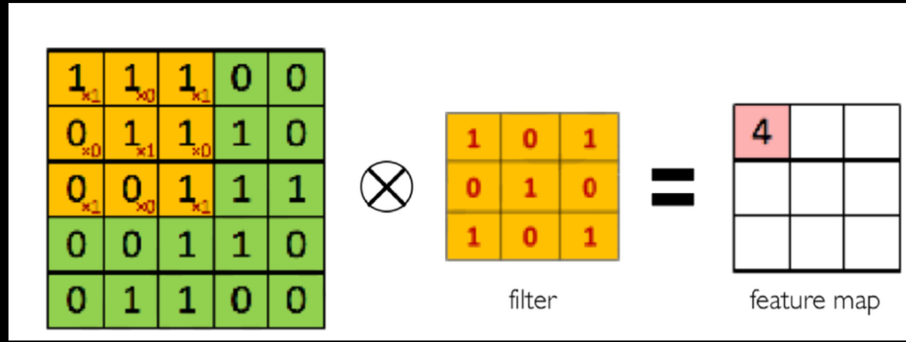


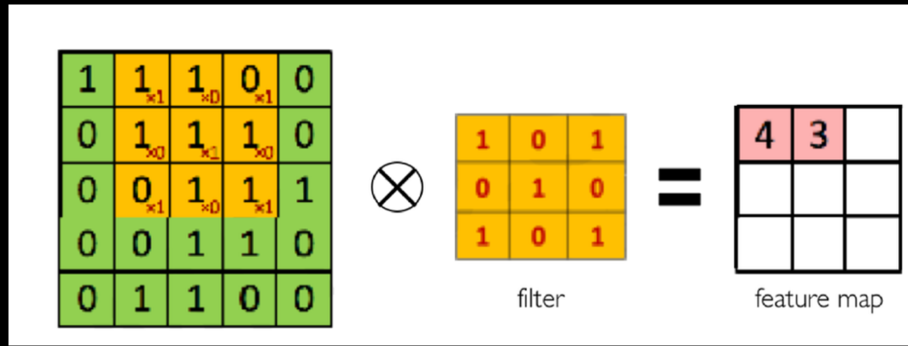


Convolutional operation

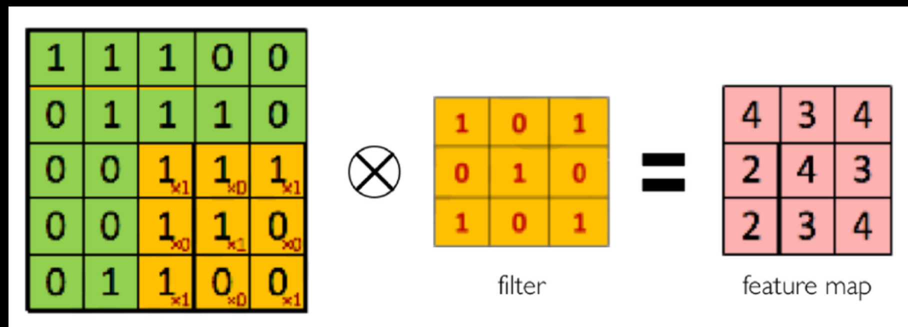






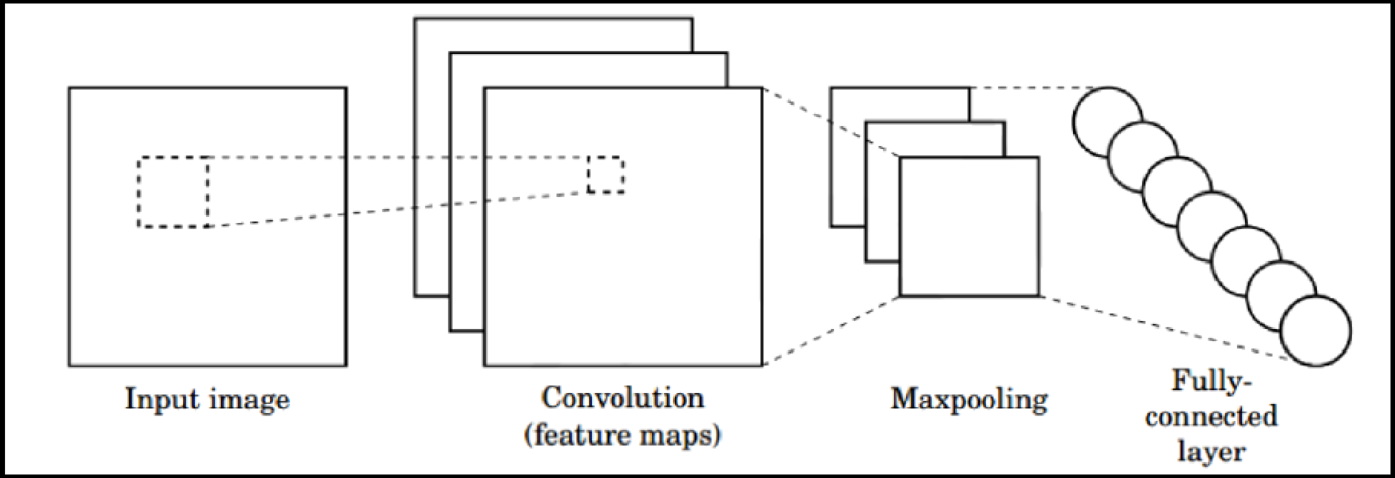


...



Convolutional neural networks (CNN)

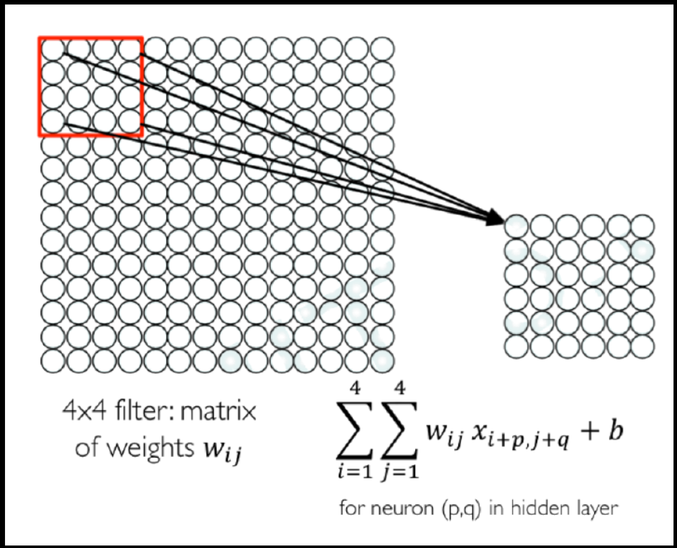
- for classification



- Convolution: apply filters to generate feature maps
- Non-linearity: often ReLU
- Pooling: Downsampling operation on each feature map

Train model with image data.
Learn weights of filters in convolutional layers.

Convolutional layer

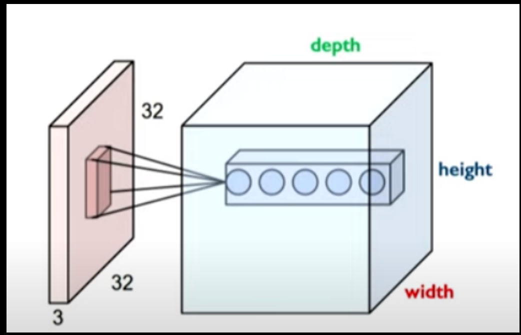


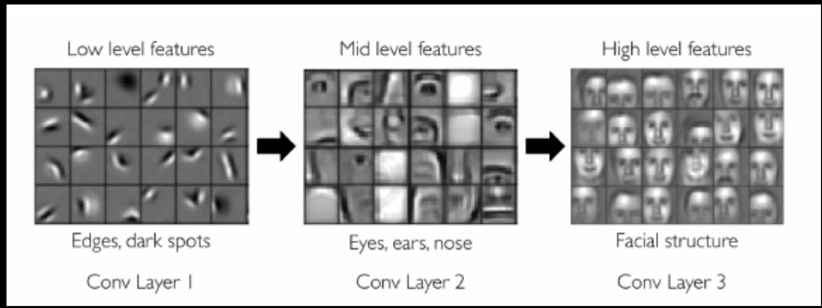
1. applying a window of weights

2. computng linear combinations

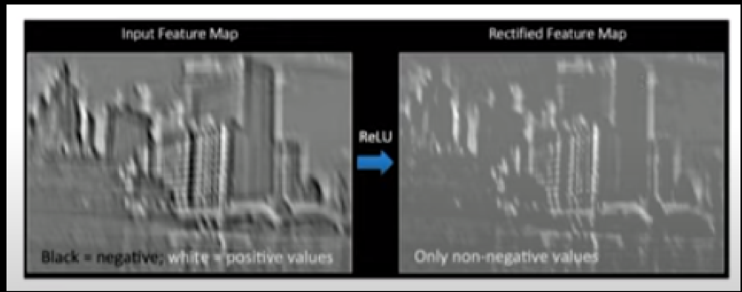
For a neuron in hidden layer

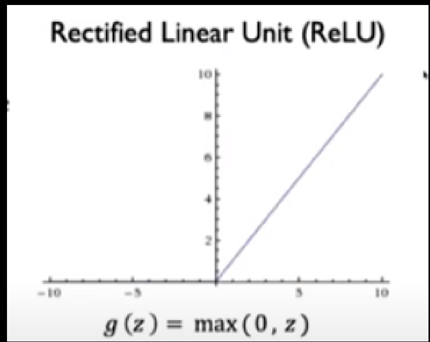
- Take inputs from patch
- Compute weighted sum
- apply bias



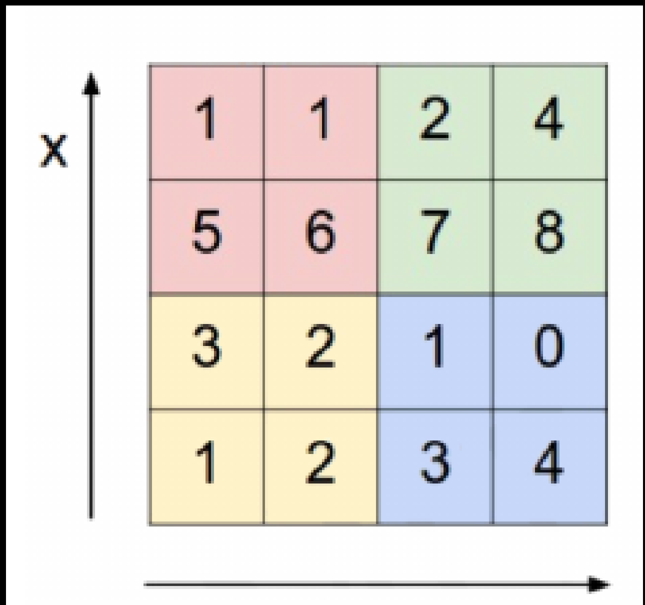


3. activating with non-linear function

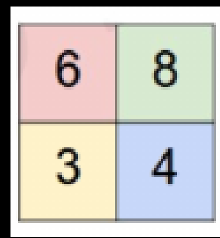




Pooling

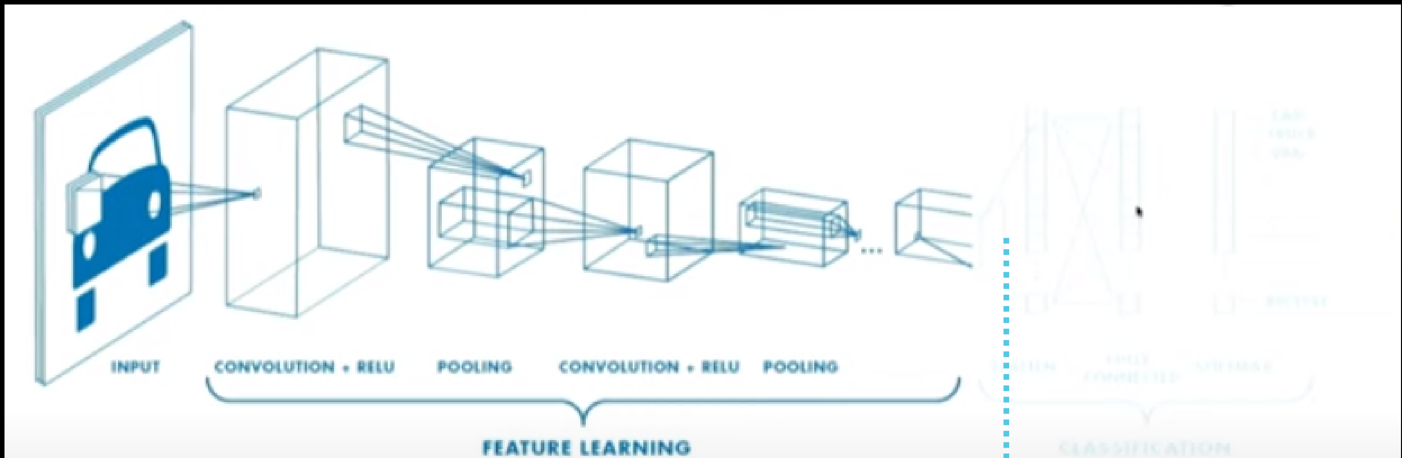


max pool with 2x2 filters
and stride 2



- Reduce dimensionality
- Spatial invariance

CNN: classification example

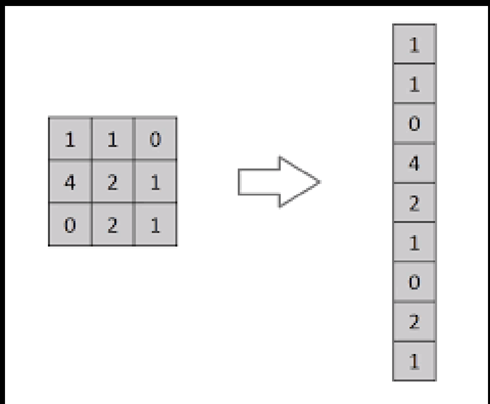


- Learns features in input image throught convolution
- Introduce non-linearity (activation func)
- Reduce dimensionality and preserve spacial invariance (pooling)

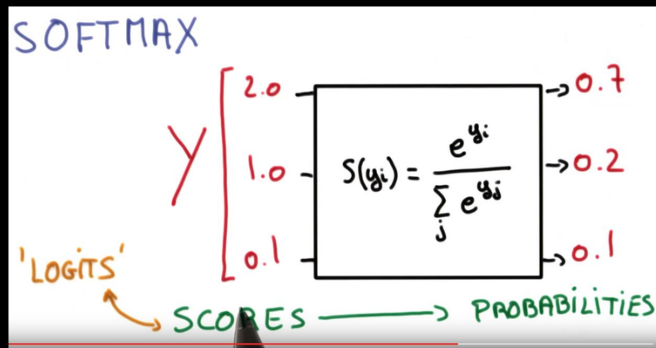
- at this point the last outpus is the i-th feature map

CNN: classification example

- CONV and POOL layers putput high-level features input
- Fully connected layer uses these features to classifying input image
- Express output as probability (image to certain class)



$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$



logits = unnormalised (or not-yet normalised) predictions (or outputs) of a model

CONICET

ML Fundamentals

Dra María Graciela Molina
FACET-UNT / CONICET
Tucumán Space Weather Center - TSWC

https://spaceweather.facet.unt.edu.ar/
IG -> @spaceweatherargentina

gmolina@herrera.unt.edu.ar