VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE | DEPARTMENT OF COMPUTER SCIENCE

Machine Learning

Artificial Neural Networks

Jan Platoš

November 22, 2023

Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VŠB - Technical University of Ostrava

Artificial Neural Networks

Artificial Neural Networks

Human nervous system

- The system is composed of cells, called neurons.
- The neurons are connected to other neurons using synapses.
- The strength of the synapses is affected by learning (external stimuli).

Artificial Neural Networks

- The system is composed of nodes, called neurons.
- These neurons are units of computation that:
    - Receive the inputs from other neurons.
    - Processes these inputs (computes).
    - Set its output.
- The computation process is affected by the input weights and activation function.
- The weights are analogous to the strength of the synapse.
- The weights are affected by the learning process.

Artificial Neural Networks

- The neural networks ability to learn is based on the architecture of the network.
    - Single-layer neural network.
    - Multi-layer neural network.
    - Recurrent neural networks.
    - Kohonen Maps (Self Organized Maps).
    - Convolution networks.
    - Deep neural networks.
    - ...
    -
- The learning is done by presenting the test instances to the network and correction of the output according to the expected output by weight adjusting.

Single-layer Neural Network: The Perceptron

- The basic architecture of neural network.
- The structure has two layers.
    - The input layer has one node for each input attribute.
    - The input node only transmit the input value to the output node.
    - The connection between input and output nodes are weighted.
    - The output layer consist of one output neuron.
    - The output neuron computes the output value.
- The class labels are from the set of $\{-1, +1\}$.

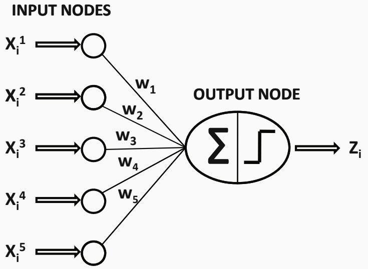


Figure 1: The Perceptron

Single-layer Neural Network

- The weighted inputs are transformed into output value.
- The value in drawn from the set $\{-1, +1\}$.
- The value may be interpreted as the perceptron prediction of the class variable.
- The weights $W = \{w_1, \ldots, w_d\}$ are modified when the predicted output does not match expected value.

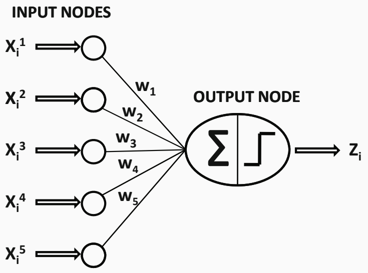


Figure 2: The Perceptron

Single-layer Neural Network

- The function learned by the perceptron is referred as *activation function*.
- The function is usually signed linear function (e.g. weighted sum).
- The $W = \{w_1, \dots, w_d\}$ are the weights for the connections of d different inputs to the output neuron.
- The d is also the dimensionality of the data.
- The b is the bias associated with the activation function.
- The output $z_i \in \{-1, +1\}$ is for the data record $\overline{X_i} = (x_i^1, \dots, x_i^d)$ computed as follows:

$$z_i = sign\left\{\sum_{j=1}^{d} w_j x_i^j + b\right\} = sign\left\{\overline{W} \cdot \overline{X_i} + b\right\}$$

Single-layer Neural Network

$$z_i = sign\left\{\sum_{j=1}^{d} w_j x_i^j + b\right\} = sign\left\{\overline{W} \cdot \overline{X_i} + b\right\}$$

- The difference between the prediction of the class value z_i and the real class value y_i is $(y_i - z_i) \in \{-2, 0, 2\}$.
- The result is 0 when the prediction and reality is the same.
- The weight vector $\overline{W}$ and bias b need to be updated, based on the error $(y_i - z_i)$.
- The learning process is iterative.
- The weight update rule for i-th input point $\overline{X_i}$ in t-th iteration is as follows:

$$\overline{W}^{t+1} = \overline{W}^{t} + \eta(y_i - z_i)\overline{X_i}$$

Single-layer Neural Network

$$\overline{W}^{t+1} = \overline{W}^{t} + \eta(y_i - z_i)\overline{X_i}$$

- The η is the learning rate that regulate the learning speed of the network.
- Each cycle per input points in the learning phase is referred as an *epoch*.
- The incremental term $(y_i - z_i)\overline{X_i}$ is the approximation of the negative of the gradient of the least=squares prediction error

$$(y_i - z_i)^2 = \left(y_i - sign\left(\overline{W} \cdot \overline{X_i} - b\right)\right)^2$$

- The update is performed on a tuple-by-tuple basis not a global over whole dataset.
- The perceptron may be considered a modified version of a gradient descent method that minimizes the squared error of prediction.

Single-layer Neural Network

$$\overline{W}^{t+1} = \overline{W}^{t} + \eta(y_i - z_i)\overline{X_i}$$

- The size of the η affect the speed of the convergence and the quality of the solution.
    - The higher value of η means faster convergence, but suboptimal solution may be found.
    - Lower values of η results in higher-quality solutions with slow convergence.
- In practice, η is decreased systematically with increasing number of epochs performed.
- Higher values at the beginning allows bigger jumps in weight space and lower values later allows precise setting of the weights.

Multi-layer Neural Network

- The perceptron, with single computational neuron produces only a linear model.
- Multi-layer perceptron adds a hidden layer beside the input and output layer.
- The hidden layer itself may consist of different type of topologies (e.g. several layers).
- The output of nodes in one layer feed the inputs of the nodes in the next layer - this behavior is called *feed-forward network*.
- The nodes in one layer are fully connected to the neurons in the previous layer.

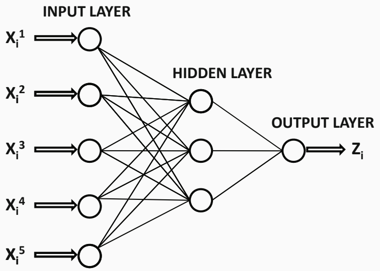


Figure 3: Multi-layer neural network

Multi-layer Neural Network

- The topology of the multi-layer feed-forward network is determined automatically.
- The perceptron may be considered as a single-layer feed-forward neural network.
- The number of layers and the number of nodes in each layer have to be determined manually.
- Standard multi-layer network uses only one hidden layer, i.e. this is considered as a two-layer feed forward neural network.
- The activation function is not limited to linear signed weighted sum, other functions such as logistic, sigmoid or hyperbolic tangents are allowed.

Multi-layer Neural Network

Sigmoid/Logistic function	$\sigma(x) = \frac{1}{1+e^{-x}}$
TanH	$\tanh(x) = \frac{(e^{x}-e^{-x})}{(e^{x}+e^{-x})}$
ReLU (Rectified linear unit)	$f(x) = \begin{cases} 0 & for x \leq 0 \\ x & for x \geq 0 \end{cases}$
Sinc	$f(x) = \begin{cases} 1 & for x = 0 \\ \frac{sin(x)}{x} & for x \neq 0 \end{cases}$
Gaussian	$f(x) = e^{x^2}$
Softmax	$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$

Multi-layer Neural Network - Learning algorithm

- The learning phase is more complicated than the one in perceptron.
- The biggest problem is the get the error in the hidden layer, because the direct class label is not defined on this level.
- Some kind of *feedback* is required from the nodes in the forward layer to the nodes in earlier layers about the *expected* outputs and corresponding errors.
- This principle is realized in the *back-propagation* algorithm.

Multi-layer Neural Network - Learning algorithm

Back-propagation algorithm

- *Forward phase:*
  - The input is fed into input neurons.
  - The computed values are propagated using current weights to next layers.
  - The final predicted output is compared with the class label and the error is determined.
- *Backward phase:*
  - The main goal is to learn weights in the backward direction by providing the error estimation from later layers to the earlier layers.
  - The estimation in the hidden layer is computed as a function of the error estimate and weight is the layers ahead.
  - The error is estimated again using the gradient method.
  - The process is complicated by the using of non-linear functions in the inner nodes.

Artificial Neural Networks - Multi-layer Neural Network - Back-propagation alg.

- Lets have an example multi-layer neural network with single output neuron.
- In each iteration do take the i-th input vector.
- Pass it through the networks using the forward pass.
- Compare the i-th output o_i to the expected value y_i.
- Compute the error and update the weight using the learning rate η.
- The goal is to optimize the weights w_i to minimize the error function of the differences between y_i and o_i.

Artificial Neural Networks - Multi-layer Neural Network - Back-propagation alg.

- The error function E over whole dataset of size n may be defined as follows:

$$E = \frac{1}{2}\sum_{i=0}^{n}(y_i - o_i)^2$$

- The weights of the neurons must be adapted according to the error produced by the neuron weight.

$$w_{i+1} = -\eta\frac{\partial E}{\partial w_i} + \mu w_i$$

Artificial Neural Networks - Multi-layer Neural Network - Back-propagation alg.

- The partial derivation may be computed using so called chain rule.

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial z} \cdot \frac{\partial z}{\partial w_i}$$

- where

$$y = \frac{1}{1 + e^{-\lambda z}} \qquad z = \sum_{i=0}^{m} w_i x_i$$

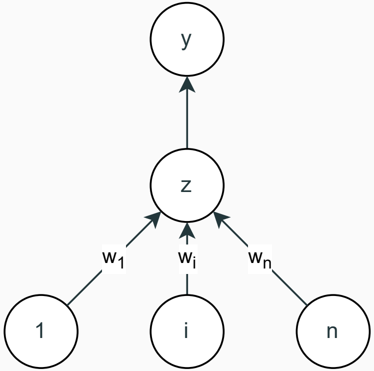
- therefore

$$\frac{\partial z}{\partial w_i} = x_i \qquad \frac{\partial y}{\partial z} = y \cdot (1 - y)\lambda$$

Artificial Neural Networks - Multi-layer Neural Network - Back-propagation alg.

- The first partial derivation computation differs for neuron from output and hidden layer.
- The solution for the output layer and *i*-th output is as follows:

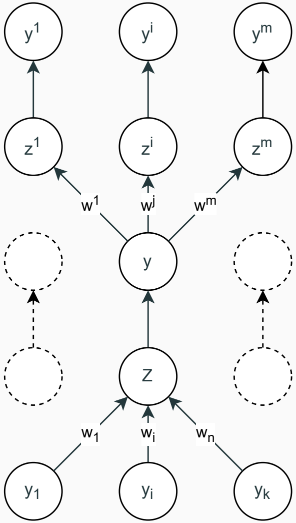
$$\frac{\partial E}{\partial y} = (y_i - o_i)$$



Artificial Neural Networks - Multi-layer Neural Network - Back-propagation alg.

- The solution for the hidden layer and *i*-th output is as follows:

$$\frac{\partial E}{\partial y} = \sum_{j=0}^{m} \frac{\partial E}{\partial z^j} \cdot \frac{\partial z^j}{\partial y} = \sum_{j=0}^{m} \frac{\partial E}{\partial z^j} \cdot w^j$$



Artificial Neural Networks - Multi-layer Neural Network

- It has ability not only to capture decision boundaries of arbitrary shapes, but also non-contiguous class distribution with different decision boundaries in different regions.
- With increasing number of nodes and layers, virtually any function may be approximated.
- **The neural networks are universal function approximate**.

Artificial Neural Networks - Multi-layer Neural Network

- This generality brings several challenges that have to be dealt with:
    - The design of the topology presents many trade=off challenges for the analyst.
    - Higher number of nodes and layers provides greater generality but also the risk of over-fitting.
    - There is very little guidance provided from the data.
    - The neural network has poor interpretability associated with the classification process.
    - The learning process is very slow and sensitive to the noise.
    - Larger networks has very slow learning process.

Multi-layer Neural Network - Learning algorithm

Other learning algorithms:

- Gradient descent
- Stochastic Gradient Descent
    - Momentum
    - Averaging
    - AdaGrad
    - RMSProp
    - Adam
- Newton’s method
- Conjugate gradient
- Quasi-Newton method
- Levenberg-Marquardt algorithm

Multi-layer Neural Network

- The multi-layer neural network is more powerful than kernel SVM in its ability to capture arbitrary functions.
- It has ability not only to capture decision boundaries of arbitrary shapes, but also non-contiguous class distribution with different decision boundaries in different regions.
- With increasing number of nodes and layers, virtually any function may be approximated.
- **The neural networks are universal function approximators**.

Multi-layer Neural Network

- This generality brings several challenges that have to be dealt with:
    - The design of the topology presents many trade=off challenges for the analyst.
    - Higher number of nodes and layers provides greater generality but also the risk of over-fitting.
    - There is very little guidance provided from the data.
    - The neural network has poor interpretability associated with the classification process.
    - The learning process is very slow and sensitive to the noise.
    - Larger networks has very slow learning process.

Convolutional Neural Network

- A version of MLP that is inspired by the visual perception of animals.
- Instead of fully connected layers it deals with the image processing with different structure.
- The layers of a CNN have neurons arranged in 3 dimensions: width, height and depth.
- The neurons inside a layer are only connected to a small region of the layer before it, called a receptive field (filters).
- CNNs exploit spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers.

Convolutional Neural Network

- Stacking many layers leads to non-linear ”filters” that become increasingly ”global”.
- This allows the network to first create good representations of small parts of the input, then assemble representations of larger areas from them.
- Each filter is replicated across the entire visual field with the same parametrization (weight vector and bias) and form a feature map.
- Features are detected regardless of their position in the visual field.

Convolution Neural Network Model

A convolution is defined as the integral of the product of the two functions after one is reversed and shifted. It is a mathmematical way how to analyze behavior of the functions and the relation between the functions.

Convolution Neural Network Model

A convolution is defined as the integral of the product of the two functions after one is reversed and shifted. It is a mathmematical way how to analyze behavior of the functions and the relation between the functions.

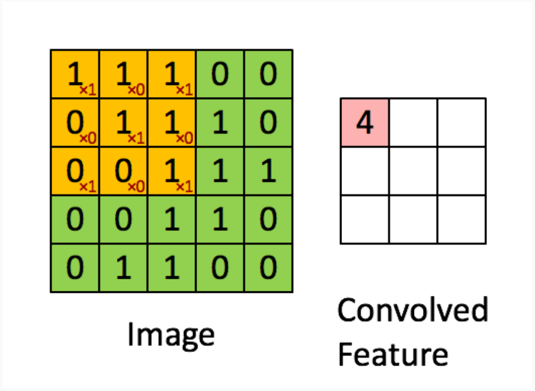
In image processing, *kernel* or *convolution matrix* or *mask* is a small matrix. In general the convolution in image processing is defined as:

$$g(x,y) = \omega * f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t)$$

where $g(x,y)$ is filtered image, $f(x,y)$ is original image, ω if the filter kernel.

Convolution Neural Network

A kernel (also called a filter) is a smaller-sized matrix in comparison to the dimensions of the input image, that consists of real valued entries.



Convolutional Neural Network - Sample Convolution Kernels

Identity

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Sobel vertical edge detection

$$\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

Sobel horizontal edge detection

$$\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Edge detection

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Sharpen

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

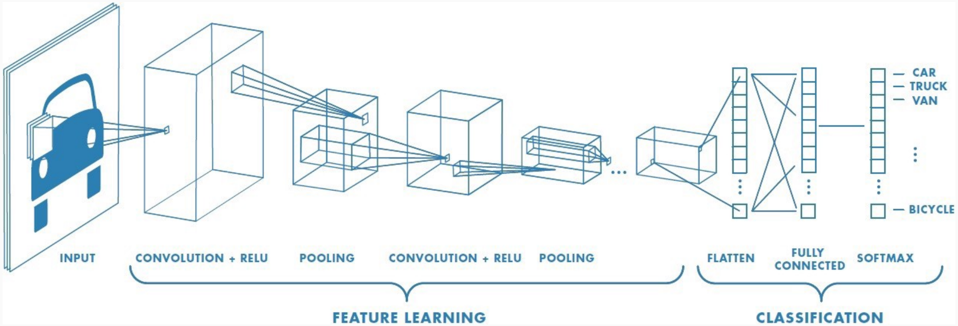
Uniform blur

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Gaussian blur 3x3

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Convolutional Neural Network



Example of the CNN architecture

Convolutional Neural Network - Basic properties

Size of the kernel defines the dimensions of the kernels.

Number of input channels reflects the number of channels of the image (grayscale, RGB, etc.)

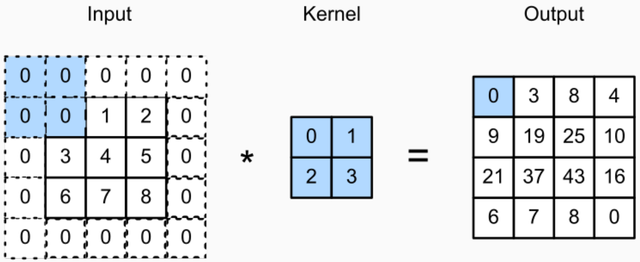
Number of output channels defines the number of kernels applied on the image, and, therefore, the output of the layer.

Stride is the size of the step that kernel is moved on the image.

Padding is system the kernel is moder on the image.

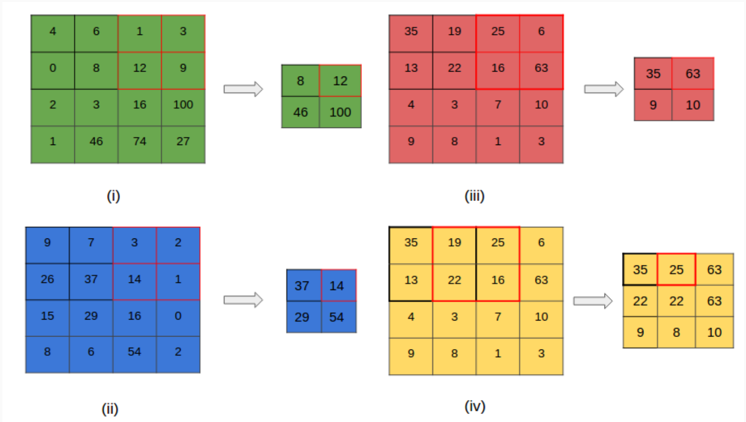
Convolutional Neural Network - Padding

One tricky issue when applying convolutional is losing pixels on the edges of our image. A straightforward solution to this problem is to add extra pixels around the boundary of our input image, which increases the effective size of the image.



Convolutional Neural Network - Pooling

Pooling is a way how to decrease the amount of information transfered from one layer to another. The standard way ho to do it is *Average Pooling* and *Maximum Pooling*.



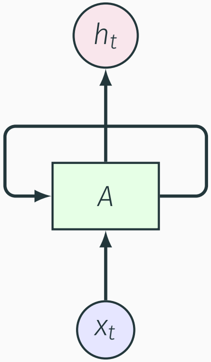
Recurrent Neural Network (RNN)

- A class of artificial neural network where connections between units form a directed cycle.
- This structure allows the dynamic temporal behavior or memory.
- Such network are able do deal with sequences of inputs as sequences not isolated inputs.
- Training is performed by gradient descent or global optimization techniques.

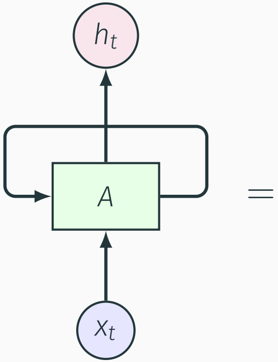
Recurrent Neural Network (RNN) - Basic types

- **Fully recurrent network** - basic architecture with recurrent connection in each level and time-varying activation function.
- **Recursive neural network** - the network that applies the same weights recursively over a graph-like structure. Designed for representation of the structures like logical terms.
- **Hopfield network** with proper learning method it is a robust content-addressable memory. Its variations it is the bidirectional associative memory.
- **Echo state network** - a special RNN that has sparsely connected random hidden layer. Only the weights of the output neuron may be changed during training.

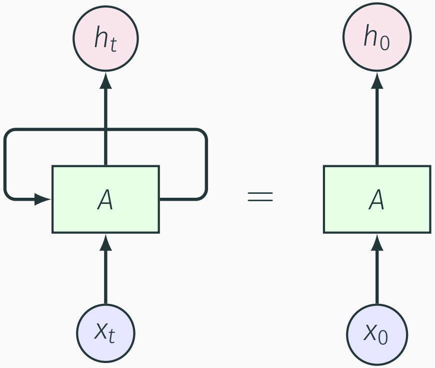
Recurrent Neural Networks



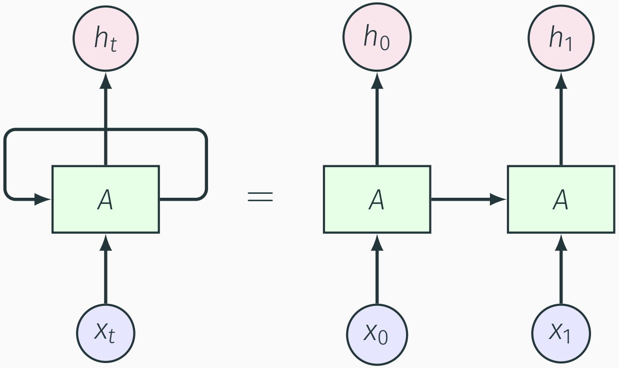
Recurrent Neural Networks



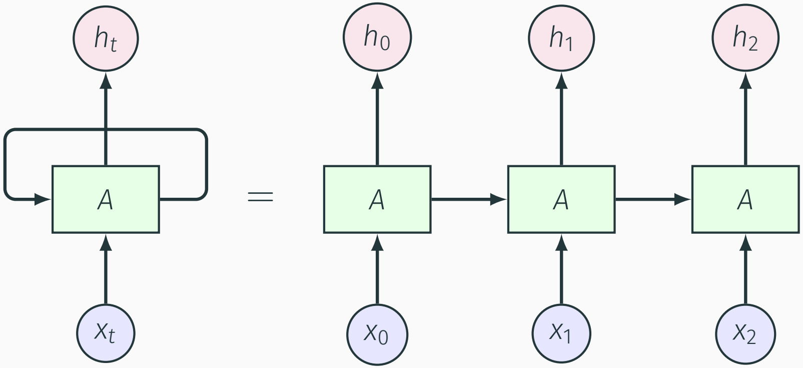
Recurrent Neural Networks



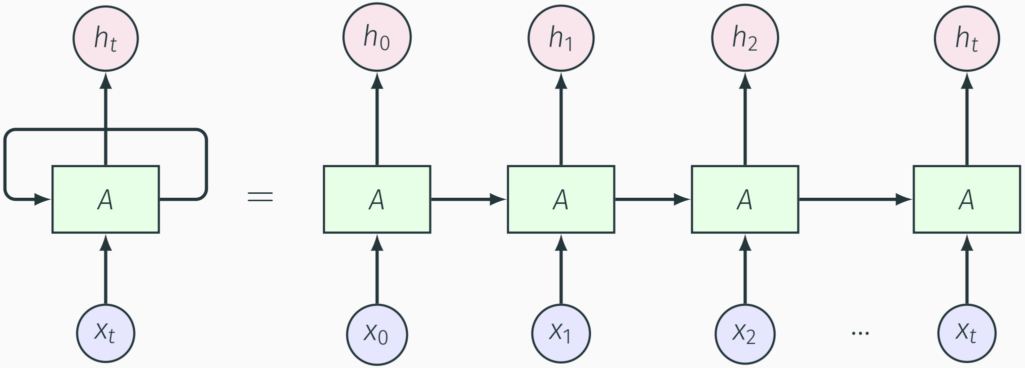
Recurrent Neural Networks



Recurrent Neural Networks



Recurrent Neural Networks



Recurrent Neural Networks

- The hidden state A is designed as h_t and its formula is as follows:

$$h_t = f(h_{t-1}, x_t)$$

- h_{t-1} is the previous state and x_t is the current input.
- The RNN neuron uses a non-linear activation functions.
- The Sigmoid, hyperbolic tangent, or ReLu is used.

Recurrent Neural Networks

- When an **tanh** activation is used then the formula is:

$$h_t = \mathbf{tanh}(W_h h_{t-1} + W_x x_t)$$

- W_h is the weight of the recurrent neuron and W_x is the weight of the input neuron.

$$y_t = W_y h_t$$

- w_y is the output weight and y_t it the current output.

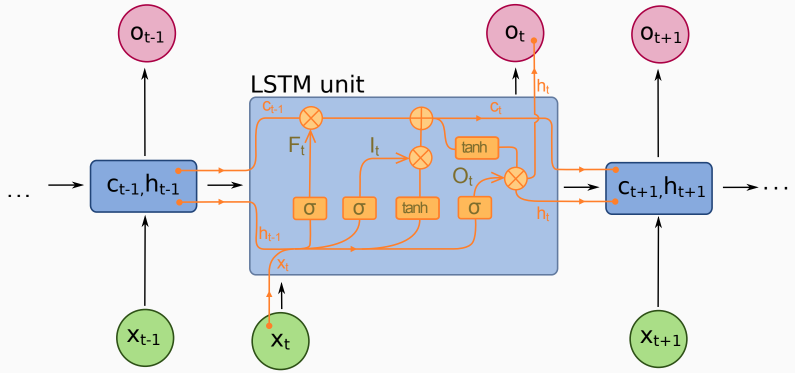
Recurrent Neural Networks - Training phase

1. A single time step of the input is provided to the network.
2. The current state using set of current input and the previous state is computed.
3. One can go as many time steps according to the problem and join the information from all the previous states.
4. Once all the time steps are completed the final current state is used to calculate the output.
5. The output is then compared to the actual output i.e the target output and the error is generated.
6. The error is then back-propagated to the network to update the weights and hence the network (RNN) is trained.

Recurrent Neural Networks

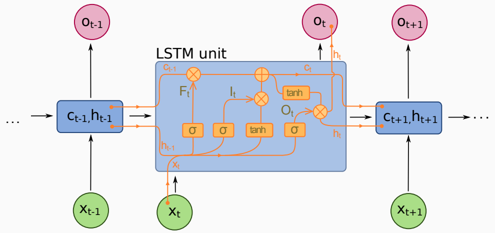
- Advantages of the RNNs:
  - An RNN remembers each and every information through time.
  - Model size does not increasing with the input length.
  - Computation of current step can use information from many steps back.
  - Weights are shared across time steps.
  - Recurrent neural network are even used with convolutional layers to extend the effective pixel neighborhood.
- Disadvantages of the RNNs:
  - Training an RNN is a very difficult task.
  - It cannot process very long sequences using *tanh* or *ReLU* as an activation function.
  - Gradient vanishing and exploding problems.

Long Short Term-Memory (LSTM)



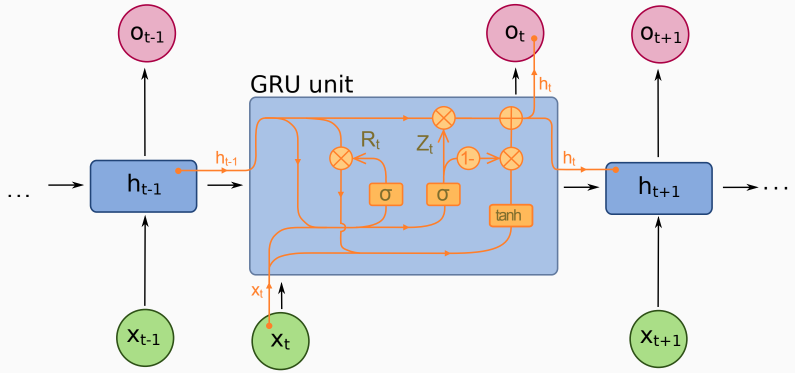
By fdeloche - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=60149410`

Long Short Term-Memory (LSTM)



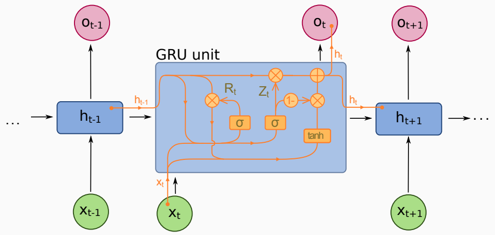
- Contains a set of recurrent connected subnets - memory blocks.
- Capable to learn long term dependencies practically.
- Gates are non-linear neural net layer (sigmoid) and regulate the amount of information that is let through.
- Each block contains one or more self-connected memory cells and three multiplicative units - represents **write=input**, **read=output** and **reset=forget** operations

Gate Recurrent Unit (GRU)



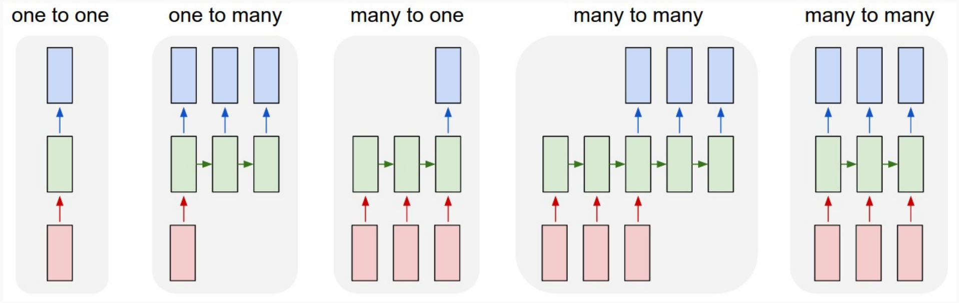
By fdeloche - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=60466441`

Gate Recurrent Unit (GRU)



- Similar to LSTM.
- Merges cell state and hidden state into one state.
- Combines the **forget** and **input** gate into an **update** gate.
- Therefore, has less parameter and less complex structure.

Types of Forecasting



A. Karpathy, `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`

Types of models

- Vanilla RNN
    - A single RNN layer and one dense layer are stacked.
    - Single layer may have as many neurons as needed.
- Stacked RNN
    - A multiple RNN layers and one dense layer are stacked.
    - Allows more complex model to be build.
- Bidirectional RNN
    - A time series is processed from both direction.
- CNN RNN
    - Convolution network detect patterns in a series.
    - These patterns are processed by the RNN layer.
- ConvRNN
    - Convolution is a direct input to the RNN neurons.

Questions?