Universitatea Tehnică a Moldovei

Ministerul Educației, Culturii și Cercetării al Republicii Moldova

Facultatea Calculatoare, Informatică și Microelectronică

Departamentul Informatică și Ingineria Sistemelor

Lucrare de laborator Nr. 2

Sisteme de inteligență artificială  
Tema : Reprezentarea cunoștințelor

A efectuat: std. gr. IA – 182 Munteanu Igor

A verificat: asist. Univ. Stratulat Stefan

Chișinău, 2020

**Scopul lucrarii:**

Proiectarea și implementarea unui neuron în 3 generații.

* Neuron de tip McCulloch-Pitts
* Neuron Recurent
* Neuron Spiking

**Sarcina de lucru:**

### descrierea caracteristicilor (rețelelor) neurale de tip Perceptron, Recurente și Spiking;

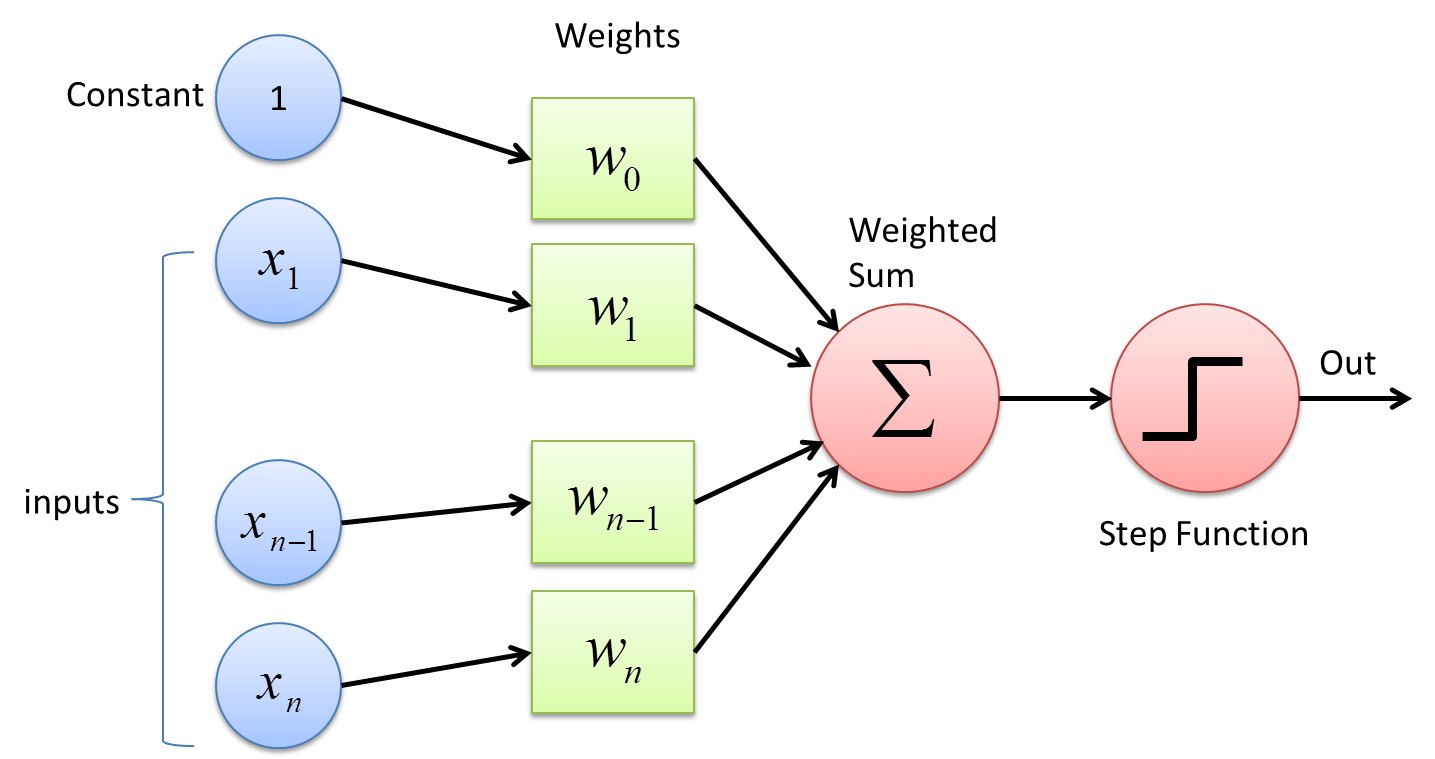
### prezentarea unor funcții de activare pentru neuroni;

### descoperirea parametrilor unor neuroni care aproximează funcții logice elementare și reprezentarea regiunilor de decizie ale acestora

***1.Neuron de tip Perceptron:***

Perceptron este un model matematic sau model computerizat de percepție a informației de către creier (un model cibernetic al creierului), propus de Frank Rosenblatt în 1957 și implementat pentru prima dată sub forma unei mașini electronice „Mark-1 “ în 1960. Perceptronul a devenit unul dintre primele modele de rețele neuronale, iar Mark-1 a devenit primul neurocomputer din lume.

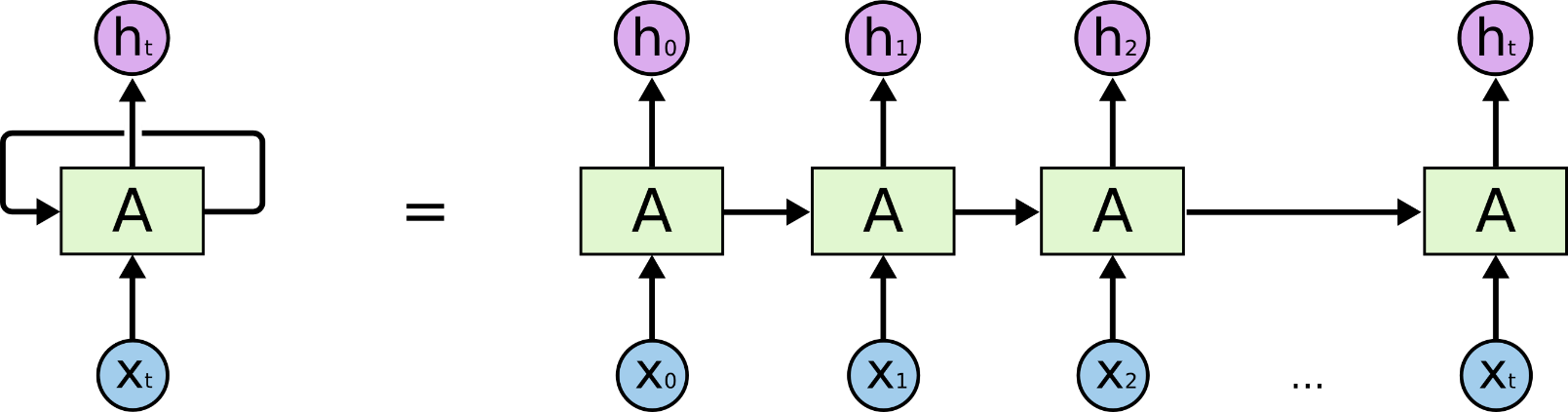
Perceptronul constă din trei tipuri de elemente și anume: semnalele provenite de la senzori sunt transmise către elementele asociative, apoi către elementele care reacționează. Astfel, perceptronii fac posibilă crearea unui set de „asociații” între stimulii de intrare și răspunsul de ieșire necesar. Din punct de vedere biologic, aceasta corespunde transformării, de exemplu, a informațiilor vizuale într-un răspuns fiziologic de la neuronii motori. Conform terminologiei moderne, perceptronii pot fi clasificați ca rețele neuronale artificiale:

* cu un singur strat ascuns;
* cu o funcție de transfer prag;
* cu propagare directă a semnalului.

***Neuroni de tip Recurente:***

Rețeaua neuronală recurentă (RNN) este un tip de rețea neuronală în care conexiunile dintre elemente formează o secvență direcționată. Acest lucru face posibilă procesarea unei serii de evenimente în timp sau lanțuri spațiale succesive. Spre deosebire de perceptronii multistrat, rețelele recurente își pot folosi memoria internă pentru a procesa secvențe de lungime arbitrară. Prin urmare, RNN-urile sunt aplicabile în sarcini în care ceva întreg este împărțit în părți, de exemplu: recunoașterea scrisului de mână sau recunoașterea vorbirii. Multe soluții arhitecturale diferite au fost propuse pentru rețelele recurente, de la simple la complexe.

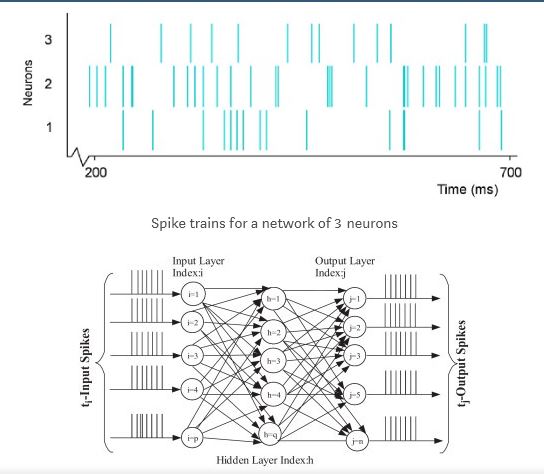
Recent, cele mai răspândite sunt rețeaua cu memorie pe termen lung și scurt (LSTM) și unitatea recurentă controlată (GRU).



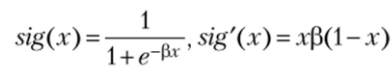
***Neuroni de tip Spiking***

Rețeaua neuronală spiking (Spiking Neural Network sau SNN) este a treia generație de rețele neuronale artificiale (ANN), care diferă de rețelele neuronale artificiale binare (prima generație) și frecvența / viteza (a doua generație) prin faptul că neuronii schimbă neuroni scurți (biologici neuroni - aproximativ 1-2 ms) prin impulsuri de aceeași amplitudine (în neuroni biologici - aproximativ 100 mV). Este cel mai realist model de rețelele neuronale artificiale fiziologice.

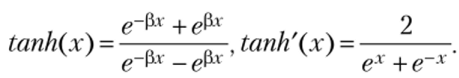
Rețeaua primește o serie de impulsuri la intrări și emite impulsuri la ieșire. În fiecare moment, fiecare neuron are o anumită valoare (analogă cu potențialul electric al neuronilor biologici) și, dacă această valoare depășește pragul, atunci neuronul trimite un singur impuls, după care valoarea sa proprie scade la un nivel sub valoarea medie (analog al procesului de reabilitare în neuroni biologici, deci numită perioadă refractară) cu 2-30 ms. Când este dezechilibrat, potențialul neuronului începe să tindă ușor la valoarea medie. Există doar doi parametri ai legăturilor de greutate ale neuronului de impuls - timpul de întârziere și greutatea.

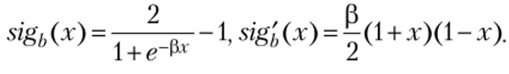


***2.Funcții de activare a neuronilor:***



1. Funcția sigmoidală



1. Funcție tangentă hiperbolică
2. Funcția sigmodală bipolară

***3.***

Codul Sursa pentru perceptron cu un singur hidden layer.

import numpy as np

def sigmoid(x):

# Activation function: f(x) = 1 / (1 + e^(-x))

return 1 / (1 + np.exp(-x))

class Neuron:

def \_\_init\_\_(self, weights, bias):

self.weights = weights

self.bias = bias

def feedforward(self, inputs):

# Input data weights, adding offsets

# using activation function

total = np.dot(self.weights, inputs) + self.bias

return sigmoid(total)

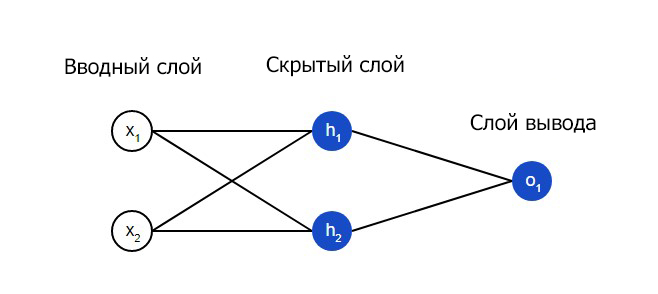
weights = np.array([0, 1]) # w1 = 0, w2 = 1

bias = 4 # b = 4

n = Neuron(weights, bias)

x = np.array([2, 3]) # x1 = 2, x2 = 3

print(n.feedforward(x)) # 0.9990889488055994



Codul sursa pentru neuron recurrent

Un exemplu simplu de model „secvențial” care procesează secvențe de numere întregi,

încorporează fiecare număr întreg într-un vector 64-dimensional, apoi procesează secvența

vectori folosind un strat `LSTM`.

import numpy as np

import tensorflow as tf

from tensorflow import keras

from tensorflow.keras import layers

model = keras.Sequential()

# Add an Embedding layer expecting input vocab of size 1000, and

# output embedding dimension of size 64.

model.add(layers.Embedding(input\_dim=1000, output\_dim=64))

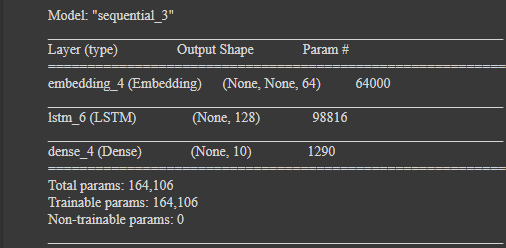
# Add a LSTM layer with 128 internal units.

model.add(layers.LSTM(128))

# Add a Dense layer with 10 units.

model.add(layers.Dense(10))

model.summary()



RNN-urile încorporate acceptă o serie de caracteristici utile:

1. Abandonare recurentă, prin intermediul argumentelor ***dropout*** și ***recurrent\_dropout***
2. Abilitatea de a procesa o secvență de intrare în sens invers, prin argumentul ***go\_backwards***
3. Derularea buclei (care poate duce la o accelerare mare atunci când se procesează secvențe scurte pe CPU), prin argumentul de ***unroll***

Codul Sursa pentru spiking neuron

Neuronul a fost testat după clasificarea binară. Poate fi extins până la orice număr de clase. Imaginile pentru două clase sunt:



Primul neuron de ieșire este activ pentru clasa 1, al doilea este activ pentru clasa 2, iar al treilea și al patrulea sunt muți pentru ambele clase. Prin urmare, înregistrând vârfurile totale ale neuronilor de ieșire, putem determina clasa căreia îi aparține tiparul.

Mai mult, pentru a demonstra rezultatele pentru clasificarea multi-clasă, simulatorul a fost testat pe următoarele 6 imagini (set de date MNIST).



from numpy import \*

import random

from matplotlib import pyplot as plt

#defining time scale

T = 50;

dt = 0.125;

time = arange(0, T+dt, dt)

#generating random spike train to be fed to neuron

S = []

for k in range(len(time)):

a = random.randrange(0,2)

S.append(a)

#initialising membrane potential vector

Pn = zeros(len(time))

#definig other parameters

Pref = 0 #resting potential

Pmin = -1 #minimum potential

Pth = 25 #threshold

D = 0.25 #leakage factor

Pspike = 4 #spike potential

count = 0 #refractory counter

t\_ref = 5 #refractory period

t\_rest = 0

#updating membrane potential according to simplified equations

for i, t in enumerate(time):

if i==0:

Pn[i] = S[i] - D

else:

if t<=t\_rest:

Pn[i] = Pref

elif t>t\_rest:

if Pn[i-1]>Pmin:

Pn[i] = Pn[i-1] + S[i] - D

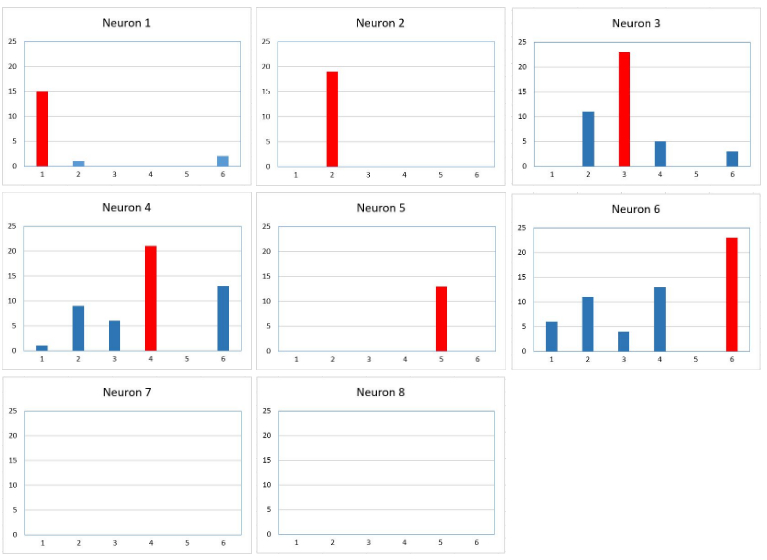
else:

Pn[i] = 0

if Pn[i]>=Pth:

Pn[i] += Pspike

t\_rest = t + t\_ref



Fiecare imagine reprezintă o clasă și fiecărei clase este delegat un neuron. 2 neuroni sunt alocați greutăți aleatorii. Iată răspunsurile fiecărui neuron la toate clasele prezentate. Axa X este numărul clasei, iar axa Y este numărul de vârfuri în timpul fiecărei simulări. Bara roșie reprezintă clasa pentru care a crescut cel mai mult.

**Concluzie:**

În urma efectuării lucrării de laborator am obținut noi cunoștințe în domeniul inteligenței artificiale.

Am aflat tipurile de rețele neuronale și funcțiile de activare ale acestora,deasemenea mi-a fost interesantă istoria apariției acestor rețele și domeniul de utilizare.

În viitor îmi va fi utilă aceată informație pentru crearea diverselor aplicații cu utilizarea inteligenței artificiale.

***B*ibliografie:**

1. [**https://github.com/Shikhargupta/Spiking-Neural-Network**](https://github.com/Shikhargupta/Spiking-Neural-Network)
2. [**https://www.tensorflow.org/guide/keras/rnn**](https://www.tensorflow.org/guide/keras/rnn)
3. **https://www.frontiersin.org/articles/10.3389/fninf.2018.00089/full**