Feed forward network with ST neurons - implementing XOR function

PAVEL STOYNOV, NIKOS MATSORAKIS
Technical university of Sofia,
Clement Ohridski 8, Sofia, 1000
BULGARIA

Abstract:—In this article we apply ST-based neurons in neural networks and check efficiency of these networks for realizing XOR functions.

Keywords:—neural networks, neurons, ST neurons, activation function.

Received: March 15, 2024. Revised: November 11, 2024. Accepted: December 14, 2024. Published: January 29, 2025.

1. Introduction

IN this article we apply ST-based neurons in neural networks and check efficiency of these networks for realizing XOR functions. The article is further development of the research in [1]–[3].

The most natural way to introduce nonlinearity into neural networks is through the nonlinear activation function of the neurons' output.

An overview of the different activation functions used in neurons is for example done by [4].

Among the most commonly used are ReLU (Rectified Linear Unit), sigmoid, hyperbolic tangent, Swish [5], [6] and others.

If n_{kt} is the internal output (the output before applying the nonlinearity) of the neuron k at time t, then the ReLU function has the form

$$L(n_{kt}) = \max(0, n_{kt}) = n_{kt}^{+}. \tag{1}$$

The sigmoid function has the form

$$L(n_{kt}) = \frac{1}{1+e^{-n_{kt}}} . \tag{2}$$

The hyperbolic tangent function is defined by the equality

$$T(n_{kt}) = \frac{e^{n_{kt}} - e^{-n_{kt}}}{e^{n_{kt}} + e^{-n_{kt}}} . \tag{3}$$

The Swish function has the form

$$L(n_{kt}) = \frac{n_{kt}}{1+e^{-n_{kt}}} . \tag{4}$$

The cumulative Gaussian function is given by the equality

$$\Phi(n_{kt}) = \int_{-\infty}^{n_{kt}} \frac{1}{\sqrt{2\pi}} e^{-0,5x^2} dx. \tag{5}$$

2. ST Neurons

A new proposal to use the so-called ST activation function, which uses the density of the probability distribution of switches (Switch-Time – ST) is presented in [7].

A random variable ξ with a switching distribution ST(n,ß) has a density

$$f_{\xi}(x) = \begin{cases} C(n,\beta) e^{-\beta x}(1+x)^n, \; x \ge 0 \\ 0, x < 0, \end{cases} \tag{6}$$

where $C(n,\beta)$ are normalization coefficients for which

$$C(n,\beta) = \frac{1}{I(n,\beta)} \tag{7}$$

and

$$I(n,\beta) = \frac{1}{\Gamma(1)} \int_0^{\infty} e^{-\beta t}(1+t)^n dt = \int_0^{\infty} e^{-\beta t}(1+t)^n dt. \tag{8}$$

In private cases when $n = 0,$ $n = 1$ and $n = 2$ the equalities $C(0,\beta) = \beta,$ $C(1,\beta) = \frac{\beta^2}{\beta+1}$ and $C(2,\beta) = \frac{\beta^3}{\beta^2+2\beta+2}$ apply accordingly. The first and second case are exponential and Lindley distribution correspondingly.

A random variable ξ with Lindley distribution has density

$$f_{\xi}(x) = \begin{cases} \frac{\beta^2}{\beta+1} e^{-\beta x}(1+x), \; x \ge 0 \\ 0, x < 0. \end{cases} \tag{9}$$

Using the ST distribution thus represented, an ST activation function with a threshold of zero can be introduced where

$$N_{kt} = \begin{cases} \int_0^{n_{kt}} C(n,b)e^{-bt}(1+t)^n dt, n_{kt} \geq 0, \\ 0, n_{kt} < 0. \end{cases} \quad (10)$$

This function depends on two parameters - parameter n and parameter b. Standard ST activation functions can be obtained by certain selection of these two parameters.

The double ST distribution (DST distribution) has the density function

$$s_\xi(x) = \begin{cases} \frac{1}{2}C(n,b)e^{-bx}(1+x)^n, x \geq 0, \\ \frac{1}{2}C(n,b)e^{bx}(1-x)^n, x < 0. \end{cases} \quad (11)$$

This distribution makes it possible to define non-threshold activation functions. The general appearance of the non-threshold ST activation function is:

$$N_{kt} = \int_{-\infty}^{n_{kt}} s(x)dx = \quad (12)$$

$$= \begin{cases} \frac{1}{2} + \frac{1}{2}\int_0^{n_{kt}} C(n,b)e^{-bx}(1+x)^n dx, n_{kt} \geq 0, \\ \frac{1}{2}\int_{-\infty}^{n_{kt}} C(n,b)e^{bx}(1-x)^n dx, n_{kt} < 0. \end{cases}$$

The neurons using ST and DST activation function we call ST and DST neurons correspondingly. The neurons using activation functions combining ST and DST with other activation function we call ST-based neurons.

In some of the types of ST neurons, the speed of convergence of the activation function is lower, and therefore a larger number of iterations is needed when training the neural networks.

Therefore, when a higher convergence rate is sought, ST neurons are more useful in combination with neurons with other activation functions, such that ST neurons can be used in the output layers of the network, while the hidden layers can use, for example, a hyperbolic tangent as activation function.

Another possibility is to combine an activation function of type ST with popular types of activation functions such as RELU. Thus, the activation function of a DST01LU neuron is set by the equality

$$N_{kt} = \begin{cases} n_{kt}, n_{kt} \geq 0, \\ \frac{1}{2}\int_{-\infty}^{n_{kt}} e^x dx - \frac{1}{2} = \frac{1}{2}e^{n_{kt}} - \frac{1}{2}, n_{kt} < 0. \end{cases} \quad (13)$$

DST01LU neurons and similar to them are interesting alternatives of the standard conventional neurons used today in neural networks.

3. Using neural networks with ST neurons and their derivatives for implementing XOR function

As an example of using ST neurons in conjunction with other types of neurons, we consider a neural network including hyperbolic tangent neurons in the inner layer and a DST01 neuron in the output layer to implement the XOR (exclusive or) function.

As an example of neurons that combine ST and other activation functions, we discuss the use of DST01LU neurons to implement the XOR (“exclusive or”) function.

As a benchmark, a network with hyperbolic tangent neurons as the activation function is used.

Functional model. We implement "exclusive or" (XOR) function using ST neurons and their derivatives and the effectiveness of such a network to the effectiveness of a forward network with the same structure, but different types of activation functions. The functional model is presented in Figure 1.

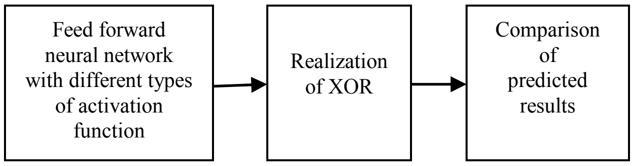
Feed forward neural network with different types of activation function
Realization of XOR
Comparison of predicted results

Fig. 1. Comparing the effectiveness of neural networks using ST-based neurons to the effectiveness of standard feed forward neural networks when implementing “exclusive or” (XOR) function. Source: the author.

Experiment scenario and structure of the experimental set-up. The experiment scenario includes the following steps:

1. Simulation of a feed forward neural network with two inputs, three neurons in the hidden layer, and one output neuron. The activation function of the neurons is different considering three cases:
   1.1. The activation function of all neurons is a hyperbolic tangent.
   1.2. The activation function of the hidden layer neurons is a hyperbolic tangent, and the output neuron is a DST01 neuron.
   1.3. All neurons in the network are DST01LU neurons.
2. The neural network is trained for 1000 iterations in each of the three cases.
3. The obtained results are analyzed.

The structure of the experimental setup is presented in Figure 2.

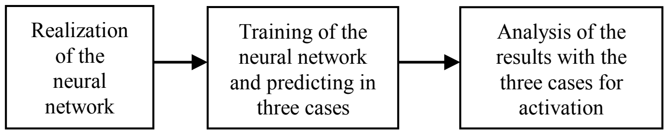
Realization of the neural network
Training of the neural network and predicting in three cases
Analysis of the results with the three cases for activation

Fig. 2. The experimental setup to test the performance of neural networks with ST-based neurons in realizing the XOR function by comparing it with the performance of conventional feed forward neural network. Source: the author.

Carrying out the experiment.

The network is implemented in the Python language and is then trained and tested for each of the three cases considered.

1. Feed forward neural network with hyperbolic tangent neurons as activation function. Python code used:

```
#xor training data
x_train = np.array([[[0,0]], [[0,1]], [[1,0]], [[1,1]]])
y_train = np.array([[[0]], [[1]], [[1]], [[0]]])
# network
net = Network()
net.add(FCLayer(2, 3))
net.add(ActivationLayer(tanh, tanh_prime))
net.add(FCLayer(3, 1))
net.add(ActivationLayer(tanh, tanh_prime))
# train
net.use(mse, mse_prime)
net.fit(x_train, y_train, epochs=1000, learning_rate=0.1)
# test
out = net.predict(x_train)
print(out)
```

The results are presented in Table 1.

2. Feed forward neural network with neurons in the hidden layer using hyperbolic tangent activation function and DST01 output neuron.

Table 1. Realization of XOR with feed forward neural network with neurons with hyperbolic tangent as an activation function. Source: the authors.

X1	X2	XOR	Value calculated by the neural network with hyperbolic tangent for activation function
0	0	0	0.00085202
0	1	1	0.97670355
1	0	1	0.97801807
1	1	0	0.00161211

The following code in Python is used:

```
#xor training data
x_train = np.array([[[0,0]], [[0,1]], [[1,0]], [[1,1]]])
y_train = np.array([[[0]], [[1]], [[1]], [[0]]])
# network
net1 = Network()
net1.add(FCLayer(2, 3))
net1.add(ActivationLayer(tanh, tanh_prime))
net1.add(FCLayer(3, 1))
net1.add(ActivationLayer(DST01, DST01_prime))
# train
net1.use(mse, mse_prime)
net1.fit(x_train, y_train,
epochs=1000, learning_rate=0.1)
# test
out1 = net1.predict(x_train)
print(out1)
```

The results are presented in Table 2.

Table 2. A feed forward network for implementation of XOR function with neurons with hyperbolic tangent activation function in the inner layer and DST01 neurons in the output. Source: the authors.

X1	X2	XOR	Value from a neural network with DST01 neuron in the output layer
0	0	0	0.000399618
0	1	1	0.96248401
1	0	1	0.96640301
1	1	0	0.01713484

3. Feed forward neural network with DST01LU neurons. Python code used:

```
#xor training data
x_train = np.array([[[0,0]], [[0,1]], [[1,0]], [[1,1]]])
y_train = np.array([[[0]], [[1]], [[1]], [[0]]])
# network
net2 = Network()
net2.add(FCLayer(2, 3))
net2.add(ActivationLayer(DST01LU,
DST01LU_prime))
net2.add(FCLayer(3, 1))
net2.add(ActivationLayer(DST01LU,
DST01LU_prime))
# train
net2.use(mse, mse_prime)
net2.fit(x_train, y_train,
epochs=1000, learning_rate=0.1)

# test
out2 = net2.predict(x_train)
print(out2)
```

The results are presented in Table 3.

Analysis of results and conclusions.

Comparing the results in Table 1, Table 2 and Table 3 shows that DST01 neurons can be used in combination with conventional neurons. Also, DST01LU neurons produce results comparable to those of conventional neurons with hyperbolic tangent as the activation function.

Table 3. Realization of XOR using feed forward network with DST01LU neurons. Source: the author.

X1	X2	XOR	Value from neural network with DST01LU neurons
0	0	0	0.000000055
0	1	1	1.000000100
1	0	1	1.000000390
1	1	0	-0.000000024

The main conclusions of this experiment are:

1. Neural networks with neurons that have a hyperbolic tangent activation function in their hidden layer and DST01 neurons in their outer layer successfully model an XOR function.

2. Neural networks with DST01LU neurons successfully model XOR function

4. Conclusion

Neural networks with ST-based neurons can be applied successfully for realizing XOR function. Further research could be applied to check efficiency of neural networks with ST-based neurons for solving different applied tasks.

References

[1] P. Stoynov, “Switch Time Family of distributions and processes and their applications to reflected surplus models,”. Annual of the Faculty of Economics and Business Administration, Sofia University “St. Kliment Ohridski” - Sofia, 2016, pp. 255-285.

[2] P. Stoynov, “Applications of ST Distributions to Neural Networks and Regression Models,” Proceedings of Eighth International Conference on New Trends in the Applications of Differential Equations in Science (NTADES’21), 6-9 September 2021, St. Constantine and Helena, Bulgaria.

[3] P. Stoynov, “Switch Time Activation Function and Stopit Regression – Some Examples,” Proceedings of XXXI International Scientific Conference Electronics-ET2022, 13-15 September 2022, Sozopol, Bulgaria.

[4] T. Szandala, Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks. arXiv. Cornell University, 2019 https://arxiv.org/ftp/arxiv/papers/2010/2010.09458.pdf

[5] P. Ramachandran, B. Zoph, V. Quoc, “Searching for activation functions,” 2017. CoRR, arXiv, vol. abs/1710.05941. https://arxiv.org/pdf/1710.05941.pdf

[6] P. Ramachandran, B. Zoph, V. Quoc SWISH: a self-gated activation function. CoRR, 2017. arXiv, https://arxiv.org/pdf/1710.05941v1.pdf?source=post_page

[7] P. Stoynov, “Switch Time Neurons. Definition and Types,” Proceedings of XXXII International Scientific Conference Electronics-ET2023, 13-15 September 2023, Sozopol, Bulgaria. Accepted for printing.