Task:

Write a Backpropagation network whose units take 3 inputs (X, Y, Z).

Answer:

In this experiment the Backpropagation network was trained in 3 variants:

1. Backpropagation network with 2nd class data;

2. Backpropagation network with 3rd class data;

3. Backpropagation network with 4th class data.

The **aim** of network trainings was to calculate and measure errors.

1. Backpropagation network with 2nd class data

ERROR OF EPOCH 1	0.139686
ERROR OF EPOCH 40	0.086214

2. Backpropagation network with 3rd class data

ERROR OF EPOCH 1	0.277152
ERROR OF EPOCH 40	0.196021

3. Backpropagation network with 4th class data.

ERROR OF EPOCH 1	0.251774
ERROR OF EPOCH 40	0.196051

Result. As can be seen, the error value is always decreasing.

b) A description of the way that you implemented the networks with enough detail to enable others to duplicate your work.

The Backpropagation network (Ognjanovski, 2019) was written. As shown on Figure 1, the network consists of units take 3 inputs (X, Y, Z).

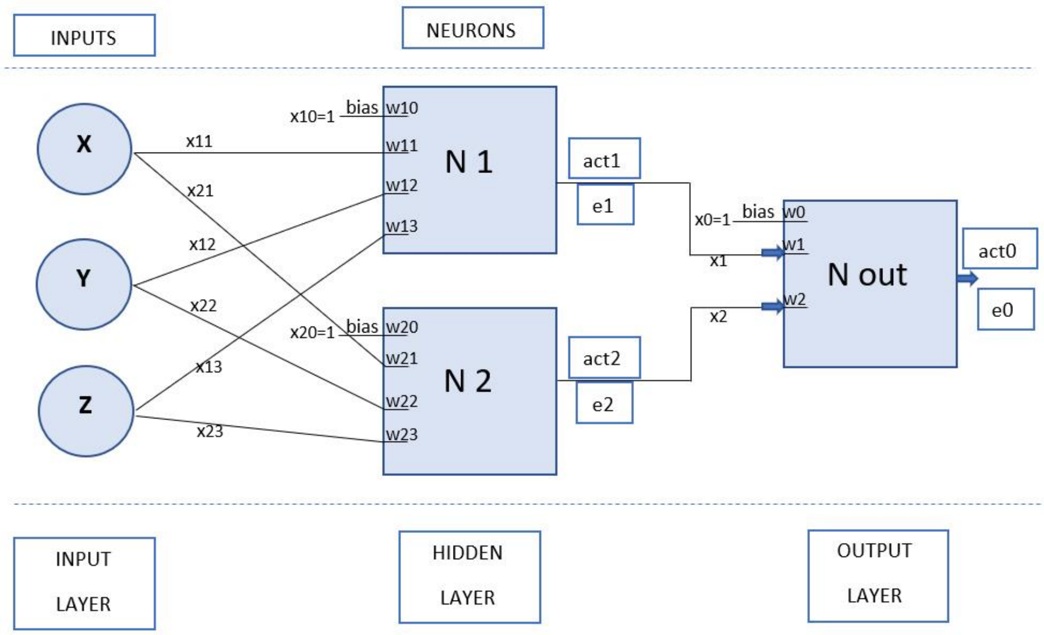

Figure 1 The 2- neurons Backpropagation network with 3 inputs (X, Y, Z)

In order to proceed with algorithm calculations, the Dataset with 3 inputs X, Y, Z was taken.

	DATASET								
Row N	X	Y	Z	2N	Taget	3N	Taget	4N	Taget
1	-0.82	0.49	0.29	1	0.8	3	0.2	3	0.2
2	-0.77	0.47	0.43	1	0.8	3	0.2	3	0.2
3	-0.73	0.55	0.41	1	0.8	3	0.2	3	0.2
4	-0.71	0.61	0.36	1	0.8	3	0.2	3	0.2
5	-0.69	0.59	0.42	1	0.8	3	0.2	3	0.2
6	-0.68	0.6	0.42	1	0.8	3	0.2	3	0.2
7	-0.66	0.58	0.48	1	0.8	3	0.2	3	0.2
8	-0.61	0.69	0.4	1	0.8	3	0.2	3	0.2
9	0.28	0.96	0.01	1	0.8	3	0.2	4	0.3
10	0.31	0.95	0.07	1	0.8	3	0.2	4	0.3
11	0.35	-0.06	0.94	2	0.6	2	0.6	2	0.6
12	0.36	0.91	-0.22	1	0.8	3	0.2	4	0.3
13	0.37	0.93	0.09	1	0.8	3	0.2	4	0.3
14	0.43	0.83	-0.34	1	0.8	3	0.2	4	0.3
15	0.43	0.9	0.01	1	0.8	3	0.2	4	0.3
16	0.47	-0.1	0.88	2	0.6	2	0.6	2	0.6
17	0.47	-0.07	0.88	2	0.6	2	0.6	2	0.6
18	0.48	0.82	-0.3	1	0.8	3	0.2	4	0.3
19	0.53	-0.23	0.82	2	0.6	2	0.6	2	0.6
20	0.54	-0.21	0.82	2	0.6	2	0.6	2	0.6
21	0.58	-0.38	0.72	2	0.6	2	0.6	2	0.6
22	0.58	0.81	0	1	0.8	3	0.2	4	0.3
23	0.65	-0.04	0.76	2	0.6	2	0.6	2	0.6
24	0.74	-0.05	0.67	2	0.6	2	0.6	2	0.6

Figure 2 The table with Winning Neurons and its Targets

Each Winning Neuron has different random Target:

N1 has T=0.8

N2 has T=0.6

N3 has T=0.2

N4 has T=0.3

The Network was created in Excel using 4 sheets:

1st sheet: Backpropagation network with 2nd class data;

2nd sheet: Backpropagation network with 3rd class data;

3rd sheet: Backpropagation network with 4th class data;

4th sheet: Backpropagation network with new testing Dataset for finding class out of 2, 3, 4 classes.

In the sheet number 4 the following changes were applied in order to find a class out.

1. Only 2 neurons were used in calculations: N1 and N2.
2. Only NET and Activation were calculated. No Error calculations.
3. Only 1 Epoch for 3 classes (3 different epochs) were calculated.

Algorithm implementation. (Ognjanovski, 2019) (missinglink.ai, 2019)

Preparation step: A. Set random Learning Rate; **B**. Set random Weights.

Algorithm:

For each Epoch **->** For each Row **->**

Step 1. Feed current row with inputs

Step2.

- Calculate Net for all neurons at the hidden layer
- Calculate Activation for all neurons at the hidden layer

Step 3. Feed an output of hidden layer

Step 4.

- Calculate Net for the neuron at the output layer
- Calculate Activation for the neuron at the output layer

Step 5. Calculate an output layer Error

Step 6. Calculate a hidden layer Errors

Step 7. Update Weights of hidden layer

Step 8. Calculate an Epoch Error.

Preparation step:

A. Set random Learning Rate = 0.9

ROW 1	LR
N1	0.9
N2	0.9
N out	0.9

Figure 3 Learning Rate

B. Set random Weights.

ROW 1	WEIGHT			
	bias	X	Y	Z
N1	0.1	0.35	0.55	0.6
N2	0.1	0.45	0.75	0.4
N out	0.1	0.2	0.3	

Figure 4 Weights

Algorithm:

For each Epoch **->** For each Row **->**

Step 1. Feed current row with inputs.

ROW 1	INPUT				WEIGHT				LR
	bias	X	Y	Z	bias	X	Y	Z	
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9
N out	1				0.1	0.2	0.3		0.9

Figure 5 Inputs

The inputs X, Y, Z from the first Row of a given Dataset was taken and put in a Row 1 Table. **Bias input** always equal 1. Dataset has 24 rows, so the same procedure was repeated for all 24 rows. 24 rows equal 1 Epoch.

Step2.

ROW 1	INPUT				WEIGHT				LR	TARGET	NET	ACT
	bias	X	Y	Z	bias	X	Y	Z				
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9	0.8	0.2565	0.563776
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9	0.8	0.2145	0.55342
N out	1				0.1	0.2	0.3		0.9	0.8		

Figure 6 Net and Activation of hidden layer

- **Calculate Net for all neurons at the hidden layer.** (missinglink.ai, 2019)

  The following formula was applied in order to calculate NET: **NET = Σ*Xi*Wi**

NET $_{N1}$ = (INPUT bias N1 * WEIGHT bias N1)+(INPUT X N1 * WEIGHT X N1)+(INPUT Y N1 * WEIGHT Y N1)+(INPUT Z N1 * WEIGHT Z N1)

NET $_{N2}$ = (INPUT bias N2 * WEIGHT bias N2)+(INPUT X N2 * WEIGHT X N2)+(INPUT Y N2 * WEIGHT Y N2)+(INPUT Z N2 * WEIGHT Z N2)

- **Calculate Activation for all neurons at the hidden layer.**

  The following formula was applied in order to calculate An Activation: $\mathbf{1/(1 + e^{-NET})}$

  $\mathbf{act}_{N1} = 1/(1 + e^{-NET\ N1})$

  $\mathbf{act}_{N2} = 1/(1 + e^{-NET\ N2})$

Step 3. Feed an output of hidden layer.

The results of $\mathbf{act}_{N1}$ and $\mathbf{act}_{N2}$ are stored as the inputs X and Y for **N out.**

ROW 1	INPUT				WEIGHT				LR	TARGET	NET	ACT
	bias	X	Y	Z	bias	X	Y	Z				
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9	0.8	0.2565	0.563776
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9	0.8	0.2145	0.55342
N out	1	0.563776	0.55342		0.1	0.2	0.3		0.9	0.8		

Figure 7 Output of hidden layer

Step 4.

- **Calculate Net for the neuron at the output layer (N out).** (Ognjanovski, 2019)

  The following formula was applied in order to calculate NET: **NET = Σ*Xi*Wi**

  NET $_{N\ out}$ = (INPUT bias N out * WEIGHT bias N out)+(INPUT X N out * WEIGHT X N out)+(INPUT Y N out * WEIGHT Y N out)+(INPUT Z N out * WEIGHT Z N out)

- **Calculate Activation for the neuron at the output layer.**

  The following formula was applied in order to calculate An Activation: $\mathbf{1/(1 + e^{-NET})}$

  $\mathbf{act}_{N\ out} = 1/(1 + e^{-NET\ N\ out})$

ROW 1	INPUT				WEIGHT				LR	TARGET	NET	ACT
	bias	X	Y	Z	bias	X	Y	Z				
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9	0.8	0.2565	0.563776
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9	0.8	0.2145	0.55342
N out	1	0.563776	0.55342		0.1	0.2	0.3		0.9	0.8	0.378781	0.593579

Figure 8 Net and Activation of output layer

Step 5. Calculate an output layer Error.

The following formula was applied in order to calculate **err** $_{\mathbf{N\ out}}$:

$$err_{N\,out} = (Target - act_{N\,out}) * act_{N\,out} * (1 - act_{N\,out})$$

ROW 1	INPUT				WEIGHT				LR	TARGET	NET	ACT	ERROR
	bias	X	Y	Z	bias	X	Y	Z					
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9	0.8	0.2565	0.5637757	
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9	0.8	0.2145	0.5534203	
N out	1	0.5637757	0.5534203		0.1	0.2	0.3		0.9	0.8	0.3787812	0.5935791	0.0497976

Figure 9 Error of output layer

Step 6. Calculate a hidden layer Errors.

The following formulas was applied in order to calculate a hidden layer **err:**

$$err_{N1} = (err_{N\,out} * W_{N1}) * act_{N1} * (1 - act_{N1})$$

$$err_{N2} = (err_{N\,out} * W_{N2}) * act_{N2} * (1 - act_{N2})$$

ROW 1	INPUT				WEIGHT				LR	TARGET	NET	ACT	ERROR
	bias	X	Y	Z	bias	X	Y	Z					
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9	0.8	0.2565	0.5637757	0.0024494
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9	0.8	0.2145	0.5534203	0.0036922
N out	1	0.5637757	0.5534203		0.1	0.2	0.3		0.9	0.8	0.3787812	0.5935791	0.0497976

Figure 10 Errors of hidden layer

Step 7. Update Weights of hidden layer.

After all calculations were done at Row 1, the Weights of Row 2 were updated respectively using formula:

W new = W old + (Learning Rate * err * X)

W new **<u>Row2</u>** **N1 bias** = W old <u>Row1</u> **N1 bias** + (Learning Rate * err <u>Row1</u> **N1** * Input <u>Row1</u> bias **N1**)

W new **N1 X** = W old **N1 X** + (Learning Rate * err **N1** * Input X **N1**)

W new **N1 Y** = W old **N1 Y** + (Learning Rate * err **N1** * Input Y **N1**)

W new **N1 Z** = W old **N1 Z** + (Learning Rate * err **N1** * Input Z **N1**)

W new **N2 bias** = W old **N2 bias** + (Learning Rate * err **N2** * Input bias **N2**)

W new N2 X = W old N2 X + (Learning Rate * err N2 * Input X N2)

W new N2 Y = W old N2 Y + (Learning Rate * err N2 * Input Y N2)

W new N2 Z = W old N2 Z + (Learning Rate * err N2 * Input Z N2)

The same procedure was done with Row 3 and all 24 rows of the Epoch 1.

EPOCH 1													
ROW 1	**INPUT**				**WEIGHT**				**LR**	**TARGET**	**NET**	**ACT**	**ERROR**
	bias	X	Y	Z	bias	X	Y	Z					
N1	1	-0.82	0.49	0.29	0.1	0.35	0.55	0.6	0.9	0.8	0.2565	0.56378	0.00245
N2	1	-0.82	0.49	0.29	0.1	0.45	0.75	0.4	0.9	0.8	0.2145	0.55342	0.00369
N out	1	0.56378	0.55342		0.1	0.2	0.3		0.9	0.8	0.37878	0.59358	0.0498
ROW 2	**INPUT**				**WEIGHT**				**LR**	**TARGET**	**NET**	**ACT**	**ERROR**
	bias	X	Y	Z	bias	X	Y	Z					
N1	1	-0.77	0.47	0.43	0.1022	0.34819	0.55108	0.60064	0.9	0.8	0.35138	0.58695	-0.0439
N2	1	-0.77	0.47	0.43	0.10332	0.44728	0.75163	0.40096	0.9	0.8	0.2846	0.57067	-0.06875
N out	1	0.58695	0.57067		0.14482	0.22527	0.3248		0.9	0.8	0.4624	0.61358	0.0442
ROW 3	**INPUT**				**WEIGHT**				**LR**	**TARGET**	**NET**	**ACT**	**ERROR**
	bias	X	Y	Z	bias	X	Y	Z					
N1	1	-0.73	0.55	0.41	0.0627	0.37861	0.53251	0.58365	0.9	0.8	0.31849	0.57896	-0.05079
N2	1	-0.73	0.55	0.41	0.04145	0.49492	0.72255	0.37436	0.9	0.8	0.23105	0.55751	-0.07579
N out	1	0.57896	0.55751		0.1846	0.24862	0.3475		0.9	0.8	0.52227	0.62768	0.04027

Figure 11 Update Weights of hidden layer

Step 8. Calculate an Epoch Error.

The following formula was applied in order to calculate an Epoch error:

Epoch error = SQRT of ((err1)^2 + (err2)^2 + (err2)^2 +…. + (err24)^2 /24)

ROW 24	**INPUT**				**WEIGHT**				**LR**	**TARGET**	**NET**	**ACT**	**ERROR**
	bias	X	Y	Z	bias	X	Y	Z					
N1	1	0.74	-0.05	0.67	-1.4115	0.106545	-0.09996	0.01632	0.9	0.6	-1.31672	0.211364	-0.06586
N2	1	0.74	-0.05	0.67	-1.71807	0.248196	-0.10096	-0.26137	0.9	0.6	-1.70447	0.153882	-0.06398
N out	1	0.211364	0.153882		0.462251	0.383793	0.480069		0.9	0.6	0.617245	0.649592	-0.01129
										ERROR OF EPOCH 1			0.139686

Figure 12 Epoch error

These eight steps were applied for 1 Epoch. In order to train this network, 40 epochs were created.

c) Table of results

	DATASET			Class out of 2 classes	Class out of 3 classes	Class out of 4 classes
Row N	X	Y	Z			
1	-0.99	0.13	0.08	1	2	2
2	-0.98	0.2	0.01	1	2	2
3	-0.96	0.29	0.03	1	2	2
4	-0.94	0.29	0.18	1	2	2
5	-0.84	0.42	0.34	1	2	2
6	-0.77	0.41	0.48	1	2	2
7	-0.73	0.57	0.37	1	2	2
8	-0.7	0.47	0.53	1	2	2
9	0.31	-0.05	0.95	1	2	2
10	0.31	-0.15	0.94	1	2	2
11	0.39	-0.3	0.87	2	2	2
12	0.41	-0.14	0.9	1	2	2
13	0.46	0.89	0.04	2	1	1
14	0.47	0.87	-0.16	2	1	1
15	0.49	0.87	0.09	2	1	1
16	0.5	-0.21	0.84	2	2	2
17	0.52	-0.28	0.8	1	2	2
18	0.56	-0.35	0.75	1	2	2
19	0.57	-0.22	0.79	2	2	2
20	0.68	0.73	0.12	1	1	1
21	0.71	0.69	0.13	1	1	1
22	0.8	0.58	0.16	2	1	1
23	0.83	0.56	-0.02	2	2	2
24	0.84	0.54	0.1	1	1	1

Figure 13 Table of testing data results

d) A commentary on the results that you have obtained, describing any features that you feel are notable

In a class out of 2 classes, the winning neuron 1 has appeared more often than neuron 2. 16 out of 24.

In a class out of 3 classes, the winning neuron 2 has appeared more often than neuron 1. 17 out of 24.

In a class out of 4 classes, the winning neuron 2 has appeared more often than neuron 1. 17 out of 24.
The class out of 3 and 4 classes has absolutely the same result in winning neurons, besides the fact that the inputs are different.

References

missinglink.ai, 2019. *Backpropagation in Neural Networks: Process, Example & Code,* Tel Aviv: s.n.

Ognjanovski, G., 2019. *Everything you need to know about Neural Networks and Backpropagation — Machine Learning Easy and Fun,* s.l.: s.n.