Artificial Neural Networks

Multilayer Perceptron

Homework #2

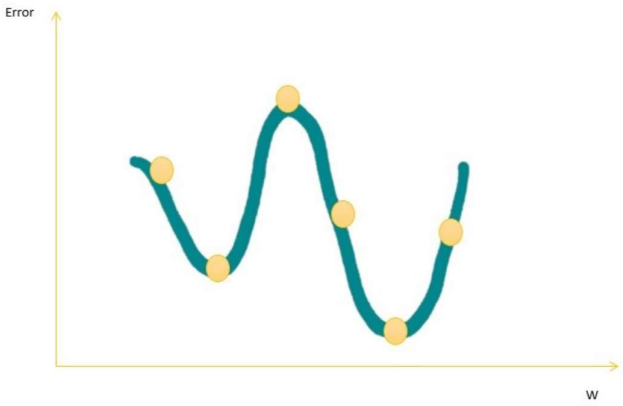
Poorya MohammadiNasab

(400722138)

Problem 1

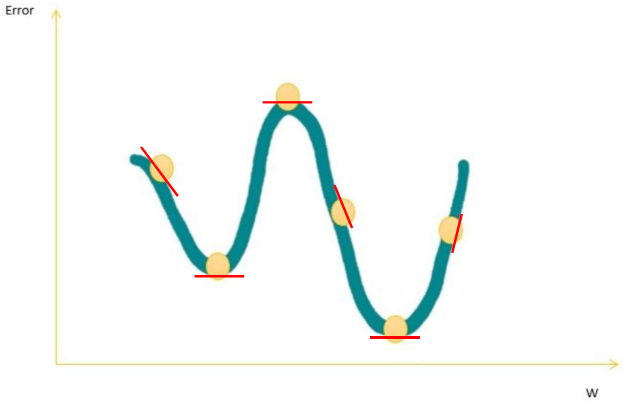
Problem 1.1

According to the figure, answer the following questions.



Problem 1.1.a

Draw the slope at the specified points and explain how the weight should change to reach the Global minimum position (according to SGD) at each point.



We can use a learning rate scheduler set to an initial high value and then gradually decrease our learning rate over time so that we don't end up bouncing off from a minimum. If the learning rate is reduced too quickly, you may get stuck in a local minimum, or even end up halfway to the minimum. If the learning rate is reduced too slowly, you may jump around the minimum for a long time and end up with a suboptimal solution assuming you halt training too early.

Problem 1.1.b

What are points number 2 and 3 called? (The points are numbered from left to right.) Since the slope is zero in these points, the optimizer may get stuck in such positions. Suggest a solution to this problem.

Point #2 is a **Local Minimum** and point #3 is a **Global Maximum.** In point #3 the error of the network is maximum. In point #2 the error rate is low, but there is another point (#5) that has a better performance. To solve the problem of stuck in local areas, we can use some common strategies.

Momentum can help the network out of local minima. This is probably the most popular extension of the backprop algorithm; it is hard to find cases where this is not used. With momentum m, the weight update at a given time t becomes.

$$\Delta w_{ij}(t) = \mu_i \delta_i y_j + m\, \Delta w_{ij}(t-1)$$

When the gradient keeps changing direction, momentum will smooth out the variations. This is particularly useful when the network is not well-conditioned. In such cases the error surface has substantially different curvature along different directions, leading to the formation of long narrow valleys.

Learning Rate Adaptation is a simple heuristics to arrive at reasonable guesses for the global and local learning rates. It is possible to refine these values significantly once training has commenced, and the network's response to the data can be observed. We will now introduce a few methods that can do so automatically by adapting the learning rates during training.

Problem 2

In neural networks, both shallow and deep networks can be used to approximate the performance of functions. Explain how these networks can estimate the best. Moreover, please define shallow and deep neural networks.

Shallow neural networks consist of only 1 or 2 hidden layers. Understanding a shallow neural network gives us an insight into what exactly is going on inside a deep neural network. A **deep neural network (DNN)** can be considered as stacked neural networks, i.e., networks composed of several layers.

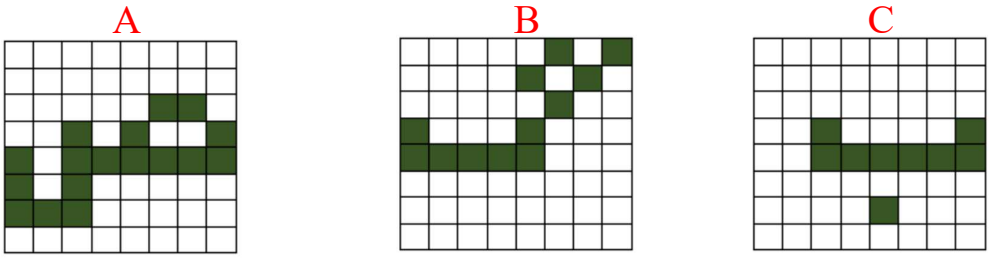
Function Approximation is describing the behavior of complex function by ensembles of simpler functions. The methods included polynomial approximation by Gauss, series expansion to compute an approximation of a function around the operating point, like the Taylor Series, and many more. The **feedforward networks** provide a universal system for representing functions in the sense that, given a function, there exists a feedforward network that approximates the function. This says that there exists a large network that approximates the function under consideration.

These were some of the attempts to understand the basics of the universal approximation power of neural networks, which makes possible the deep learning field to be effective on a wide range of tasks.

- Training a neural network on data approximates the unknown underlying mapping function from inputs to outputs.
- Problems such as Over-fitting, while training the neural network hinders the results over new data (unseen).

Problem 3

In the following figure, you can see the patterns of some Persian letters. First, write down each pattern as a 1 × 64 vector. The vectors are called transposed features. Then suggest an MLP network that can separate these patterns. Consider the white squares as 0 and the colored ones as 1. (Hint: To make the problem simple, you can remove the common features among the vectors (those with similar values).)



A

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0	1

1	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

B

0	0	0	0	0	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0

1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

C

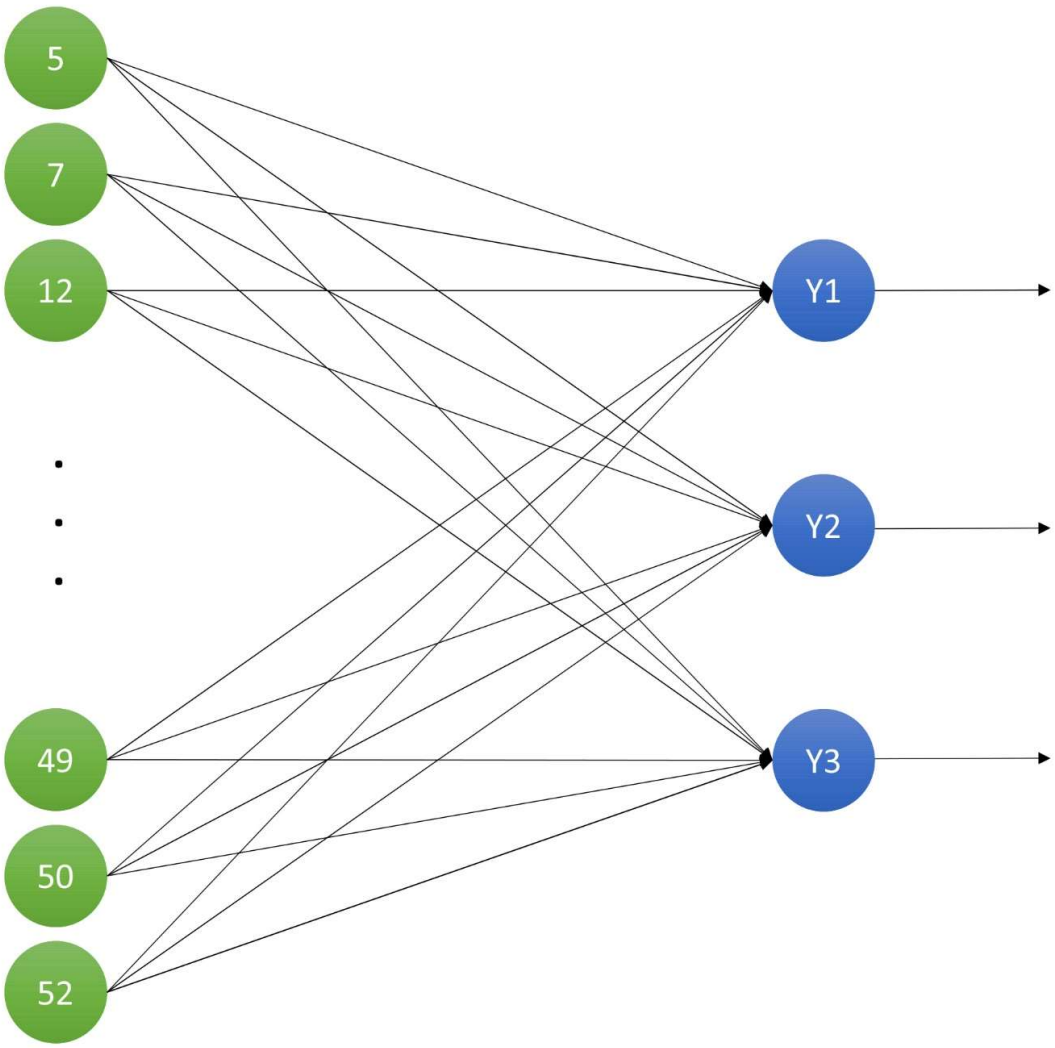
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1

0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Here we have a sample which shows the pixels that are always 0 (colored as blue).

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

Now we can see that we have only 24 pixels which are not zero and we can use them as network inputs. I suggest a single layer network with 24 inputs and 3 outputs. Figure below illustrates the overall diagram of the proposed network.



References

1) http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/

2) https://medium.com/@divakar_239/stochastic-vs-batch-gradient-descent-8820568eada1

3) https://cnl.salk.edu/~schraudo/teach/NNcourse/momrate.html

4) https://medium.com/analytics-vidhya/gradient-descent-and-beyond-ef5cbcc4d83e

5) https://openreview.net/forum?id=BylIciRcYQ

6) https://towardsdatascience.com/shallow-neural-networks-23594aa97a5

7) https://www.sciencedirect.com/topics/computer-science/deep-neural-network

8) https://towardsdatascience.com/exploring-neural-networks-and-their-fascinating-effectiveness-81ebc054cb16