دانشگاه صنعتی خواجه نصیرالدین طوسی

تمرین سری اول درس یادگیری عمیق

علی باقری‌نژاد

۴۰۲۰۲۸۵۴

Contents

۱. سوال اول

۲-۱- شبکه عصبی یک لایه

The goal of this network is to recognize the target number for different inputs. That is, we should arrange and choose the wights in such way that if out input was a number between 6 to 9, the largest possible value would be displayed in the corresponding output, but the output would show a smaller value that the threshold if the input is not any of our desired numbers.

On this basis, we put the number 1 for the corresponding weights of the inputs that are on when displaying the number and we put the number -1 for the corresponding weights of inputs that are off when displaying the number. The negativity is because if a wrong element is on, a large error would be given to the output to avoid detecting the wrong number. Finally, the weight matrix is divided by 7 so that the output is normalized and the value would never be less than 0 or bigger than 1. S

$$W=\frac{1}{7}\begin{bmatrix}1 & -1 & 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & -1 & -1 & -1 & -1\\ 1 & 1 & 1 & 1 & 1 & 1 & 1\\ 1 & 1 & -1 & 1 & 1 & 1 & 1\end{bmatrix}$$

But it shout be kept in mind that all the input values should be either 0 or 1, which whether the corresponding element in the LED display is on or off.

۱-۳ شبکه عصبی دولایه

۱-۳-۱ الف

For this mode, we determine the objective of each layer of the network:

Middle layer (hidden layer): each of the neurons of this layers must count the number of the lit LEDs of one of the two different modes. (mode one, all the elements on the right and the mid element and mode two, all other elements that are not in mode one)

Output layer: recognizing the displayed number

براساس تعریف وضایف هر لایه، وزن های هر لایه را می یابیم.

Based on the definition of the functionality of each layer, we find the weights for each layer.

Middle layer: As defined, the functionality of the neurons of this layer is to count light elements of one of the defined modes. So, the corresponding weight for each element is 1 or 0. (if the element exists in the defined mode, its weight is 1, otherwise it is 0)

$$W^{[1]} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Output layer: This layer should determine the input number based on the output of the previous layer. For this purpose, we should determine the weights so that the neuron responsible for each number, has the maximum value when that number is given to the network.

$$W^{[2]} = \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} \\ 1 & -1 \\ \frac{1}{7} & \frac{1}{7} \\ \frac{1}{2} & -\frac{1}{6} \end{bmatrix}$$

By applying these weights to the output layer, if the output of each neuron is greater-equal than 1, then that number is given as the input to the network.

۲-۳-۱ ب

The detecting pattern in each layer is as follows:

Middle layer: counting the number of lit elements in two LED segmentation modes.

Output layer: distinction of the input number based on the output of the middle layer.

۳-۳-۱ ج

$Single\ layer: 4 * 7 = 28$

$Double\ layer: 2 * 7 + 4 * 2 = 14 + 8 = 22$

سوال ۲

In the first step, we import the 'Iris' data into colab environment through the scikit learn library. Then we store the data array with the target array in a pandas Dataframe object.

۲-۱ نمایش پراکندگی داده و ایجاد یک مدل Adaline

۲-۱-۱ نمایش پخش داده ها

Utilizing the functions provided by 'Matplotlib' library, we create a visual representation of the relationship between 'Sepal-width' and 'Sepal-length' charactersitics on a graph.

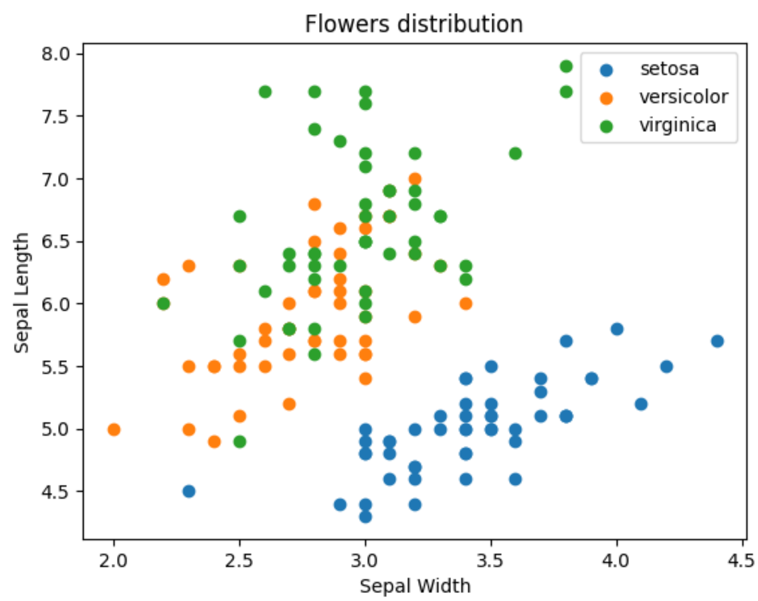

Figure ۱ Distribution of flowers based on Sepal-length and Sepal-width features

Figure 1 illunstrates the data distribution with respect to Sepal-width and Sepal-length traits.

ایجاد مدل Adaline

Adaline model is essentially a neural network model with a single neuron with a linear activation fuction. The diagram below provides a representiation of an Adaline neural network model.

Prior to constructing and training the model, we standardized the input data using the scikit learn library. This preprocessing step was undertake to improve the performance of the neural network model.

In the construction of the Adaline neural network, the numpy library was employed. Initial values of wights and bias were generated using `np.random.rand`. Subsequently, in a for loop, computations were performed and the error was calculated. It's noteworthy that the neuron's output was a continues number, necessitating a transformation into 0 or 1. This transformation enabled the comparison between the output and the target value. Facilitating the calculation of the error value.

سوال ۳

۳-۱ ایجاد مدل

I created a Autoencoder with the desired structure using Dense layers. For the encoder layers, the output of the neurons were given to a tanh antivation function after going through a relu activation function.

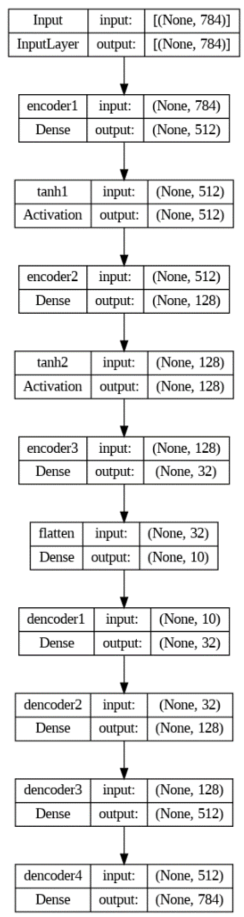

Figure ۲ Architecture of autoencodere

۳-۱ آموزش مدل

Inorther to train an autoencoder model for dimensionality reduction on the MNIST dataset, I began by loading the MNIST dataset and designing the autoencoder architecture. I then implemented a custom cost function based on the approach described in the referenced paper (Lu. S, 2022).

$$w_i = \frac{\sum_{l_p=l_q} e^{-(x_{ip}-x_{iq})^2}}{\sum_{l_p=l_q} 1} \cdot \frac{\sum_{l_p\neq l_q} (1-e^{-(x_{ip}-x_{iq})^2})}{\sum_{l_p\neq l_q} 1}$$

Figure ۳ Formula of calculating the weights for cost function(Lu. S, 2022).

Figure 3 illustrates the approach of the paper to design the weight for each pixcel. In order to reconstruct the weight for the data, the following approach was taken:

- Devide images into baches of 32
- Separating imges into 2 goups, with the same label and with different labels
- Calculating $\sum e^{-(x_{ip}-x_{iq})^2} or \sum 1-e^{-((x_{ip}-x_{iq})^2)}$ for each pixcel in the batch.(I created a matrix of all $(x_{ip}-x_{iq})$s for each group of each pixcel and then raised it to the power of 2 and continued the calculations.)
- Normalizaing the value of the weight for each of two groups.
- Multiplying the final values to calculate the weight.

To improve training efficiency, this algorithm is implemented as a Class. It takes data as input and outputs weights for all batches and pixels. Weight extraction, a previously time-consuming step, is now performed only once for all data and saved. This eliminates the need for weight extraction during future training runs.

During training, the model minimized a cost function combining MSE with the weighted loss function proposed in the paper. After training, I used the scikit-learn k-means clustering algorithm to group the model's encoded outputs.

PART 4 OF QUESTION 2 WAS NOT ANSWERED.