First Homework

Ali Gur

September 2024

1 What is deep learning, and how does it differ from traditional machine learning?

In classical machine learning, there are models such as naive bayes, decision tree, linear regression that are successful in solving certain problems and use different methodologies underneath. In deep learning, there are neural networks consisting of neurons in different layers. These neural networks can be of different sizes and depths. What we call deep learning is the approximation of these neural networks to the functions that will solve the problems we want with a large amount of data. Compared to classical machine learning, deep learning is quite successful in solving problems that seem very complex.

2 What do we mean by forward propagation?

The input vector enters the neural network and progresses through the layers, processing it and eventually giving the desired output vector.

3 What is backpropagation, and why is it essential for neural networks?

In backpropagation, gradients are calculated using the values obtained in forward propagation, actual values and loss function. These gradients are used to optimize the loss function, and different optimizers perform gradient descent in the background. In this way, appropriate parameters are obtained using the data we have. This is why backpropagation is very important for deep learning.

4 Why do we use activation functions for the layers? What happens if we don't use activation functions?

Activation functions add non-linearity by being placed between layers in neural networks. If these functions are not used, data containing complex patterns will not be learned and will result in underfitting.

5 Write forward propagation equations and corresponding dimensions of the matrices and vectors for the neural network below. Assume your batch size is 1. (Dimensions for weight matrices and input vectors for each layer, as we did in the session.)

Layer 0:

$$a^{[0]} = \begin{bmatrix} a_1^{[0]} \\ a_2^{[0]} \\ a_3^{[0]} \\ a_4^{[0]} \end{bmatrix}, \quad a^{[0]} \in R^{4\times 1}$$

Layer 1:

$$z^{[1]} = W^{[1]}a^{[0]} + b^{[1]}, \quad a^{[1]} = \sigma(z^{[1]})$$
$$W^{[1]} \in R^{6\times 4},\ b^{[1]} \in R^{6\times 1},\ a^{[1]} \in R^{6\times 1}$$

Layer 2:

$$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}, \quad a^{[2]} = \sigma(z^{[2]})$$
$$W^{[2]} \in R^{4\times 6},\ b^{[2]} \in R^{4\times 1},\ a^{[2]} \in R^{4\times 1}$$

Layer 3:

$$z^{[3]} = W^{[3]}a^{[2]} + b^{[3]}, \quad a^{[3]} = \sigma(z^{[3]})$$
$$W^{[3]} \in R^{3\times 4},\ b^{[3]} \in R^{3\times 1},\ a^{[3]} \in R^{3\times 1}$$

6 When does our model have underfitting and overfitting? How do we deter- mine these situations? Evaluate your answer.

If the model cannot discover the pattern in the data, that is, if it cannot learn the data, underfitting occurs. If the model is too simple due to reasons such as not having enough layers in the model or not using activation functions, this causes underfitting. On the other hand, if the model is too complex and has too many neurons, the model learns too much of the train data, in other words, memorizes it along with the noise, this is called overfitting. A high train loss is an indicator of underfitting. On the other hand, if the train loss is very low and the validation loss is high, overfitting has occurred.

7 What is the learning rate in the context of training neural networks? What happens if the learning rate is too high or too low?

To change the model parameters in a way that will reduce the loss function and thus optimize it, the gradient descent algorithm is used. In this algorithm, steps are taken in the opposite direction of the gradient. Here, the learning rate determines the size of the step to be taken. If the learning rate is high, the steps will be large and will oscillate instead of approaching the optimum point, so we will not get the result we want. On the other hand, if it is too small, it will take a long time to converge to the optimum point, or maybe it will not converge at all. Therefore, choosing the right learning rate by providing balance is important for training the model.

8 What is dropout, and how does it help in training neural networks?

For certain reasons, the model can overfit. Some regularization methods are used to prevent overfitting. Dropout is one of them. In dropout, some neurons in the neural networks are disabled, thus preventing the model from learning too much while training, i.e. overfitting.

9 What is the main difference between model's weights and hyperparameters?

Briefly, model parameters are the parameters that the model changes and learns while learning the data, while hyperparameters are the parameters that we change about the model. For example, weights and biases in the model are model parameters, learning rate, number of layers, type of activation function we use, optimizer, loss function, etc. are hyperparameters.

10 Explain the difference between batch gradient descent, stochastic gradient descent, and mini-batch gradient descent.

In full batch gradient descent, the entire batch is used in gradient calculations. Although more reliable and decisive steps are taken, this is slow and costly in terms of calculation since the entire batch is used. Stochastic gradient descent can be used to solve this problem. Stochastic gradient descent is based on gradient calculation and stepping by selecting random examples from the batch we have. Since there is randomness, irregular stepping and slow convergence can occur. Mini batch gradient descent is used to reduce the effect of this, where a portion of the batch we have is used.

11 Explain the difference between Adam and RMSprop optimizers.

RMSprop is an optimizer that normalizes the steps taken by calculating the root mean square values of previous gradients and thus provides adaptive learning rate. This optimizer solves the problem of vanishing and exploding gradients. In addition to the RMSprop feature, Adam optimizer also uses the momentum concept, which increases the speed of steps taken in a certain direction and provides fast convergence. Adam optimizer performs well on many neural networks. However, it can be computationally expensive.