CS 577 s21 - Assignment 2

Due by 3/9/2021

In this assignment you will implement a neural network yourself instead of using a GPU framework as in the previous assignment.

Part 1 (theoretical questions)

Artificial neurons

1. Given a neuron with linear activation and weights $(0.1, 0.2, 0.3)$ compute the output for the input vector $(1, 1)$.
2. Given a linear discriminant function $g(x)$ explain what should be its value on one side of the decision boundary, on the other side of the decision boundary, and on the decision boundary.
3. Given a linear discriminant function $g(x_1, x_2) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ explain the meaning of the coefficients $\theta_{0,}\theta_{1,}\theta_2$.
4. Let $g(x_1, x_2) = 1 + 2x_1 + 3x_2$. Find the normal to the decision boundary defined by $g(x_1, x_2)$ and the distance of the decision boundary from the origin (note that for the distance computation the normal has to be normalized).
5. When writing a discriminant function as $g(x) = \theta^T x$ where x and θ are vectors, explain where the bias coefficient in θ is and what is the required change needed to the feature vector x to allow writing the discriminant in this way.
6. Write the expressions for the step and logistic (sigmoid) activation functions. Explain the advantage of the sigmoid activation over step activation. What happens to the sigmoid when θ is small?
7. Explain how the sigmoid is obtained using a linear function to model the log-likelihood ratio.
8. Write the derivative of a sigmoid and use it to compute the derivative of log-sigmoid.
9. Explain how to compute the direction used for updating parameters in gradient descent. How is the size of the update controlled?

10. Explain the stop condition for gradient descent. Should the condition use the loss change or parameter change? Explain your answer.

11. Explain what happens when using a learning rate that is too large or too small.

12. Explain the empirical error loss is computed. What is the problem in using the empirical error loss in gradient descent?

13. Write the log likelihood for a binary classifier and show how to develop it into binary cross-entropy.

14. Write the gradient of the binary cross-entropy loss function with respect to the parameter vector when using a sigmoid activation function. Write the gradient descent update rule when using this loss function. Explain the meaning of the update equation you wrote.

15. Explain the two strategies for multi-class classification (one against all others, one against each other). In which strategy is it easier to discriminate the data?

16. Explain the template matching interpretation of linear discriminants. Using this interpretation what is the meaning of using multiple linear discriminant functions.

17. Explain the need for using softmax in the output layer of a multi-class classifier.

18. Write the derivative of softmax and use it to compute the derivative of log-softmax.

19. Write the log likelihood for a multiclass classifier and show how to develop it into categorical cross-entropy.

20. Write the gradient of the categorical cross-entropy loss function with respect to the parameter vector when using a softmax activation function. Write the gradient descent update rule when using this loss function. Explain the meaning of the update equation you wrote.

Neural networks

1. Given a two layer network where the number of hidden units is larger than the number of inputs explain the dimensionality increase interpretation of the hidden layer. Explain what is the possible benefit of such dimensionality increase.

2. Given a two layer network where the number of hidden units is smaller than the number of inputs explain the dimensionality decrease interpretation of the hidden layer. Explain what is the possible benefit of dimensionality decrease.

3. Given the update equations for the output layer in a 2 layer network, explain why we cannot use directly the same equations for the hidden layer. Assume that both layers have the same activation function.

4. Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2-layer regression network with a single output where the hidden layers use sigmoid activation and the output layer uses linear activation.

5. Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2-layer regression network with multiple outputs where the hidden layers use sigmoid activation and the output layer uses linear activation.

6. Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2-layer binary classification network where the hidden layers use sigmoid activation and the output layer uses sigmoid activation.

7. Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2-layer multiclass classification network where the hidden layers use sigmoid activation and the output layer uses softmax activation.

8. Describe how weights in the network should be initialized and explain why they are initialized in this way.

Computation graphs

1. Explain the advantage of using computation graphs. Describe what is computed during the forward pass and backward pass of the network during backpropagation. Explain what each node in the graph needs to be able to compute and what information it needs to store (and why).

2. Given a two layer network where the hidden layer output is given by $z = f_1(W_1, x)$ and the output layer output is given by $\hat{y} = f_2(W_2, z)$ express the output $\hat{y}$ as a function composition between $f_1()$ and $f_2()$. Assuming the output layer has a single node, express the L_2 loss for a batch $\{x^{(i)}, y^{(i)}\}_{i=1}^{m}$ in terms of the composite function you wrote and use the chain rule to write its derivatives with respect to W_1 and W_2. You do not need to compute the actual derivatives of $f_1()$ and $f_2()$ as they are not specified. Repeat the question with cross entropy loss instead of L_2.

3. Repeat the previous question for a multiple output network.

4. Write the computation graph for a two layer regression network with 2 nodes in the hidden layer and 1 node in the output layer. Assume sigmoid activation in the hidden layer and linear activation in the output layer. Assume the feature vectors are 3-dimensional. List all the inputs of the computation graphs and write all the derivatives needed to compute the loss gradient with respect to network parameters when using the backpropagation pass.

5. Describe the back flow patterns for the following nodes: addition of a constant, addition of two inputs, multiplication by a constant, multiplication of two inputs, max of two inputs, min of two inputs.

6. Explain what derivative you expect when the input to a node is a vector and the output is a scalar. White all the components of the derivative.

7. Explain what derivative you expect when the input to a node is a vector and the output is a vector. White all the components of the derivative.

8. Explain what derivative you expect when the input to a node is a matrix and the output is a scalar. White all the components of the derivative.

9. Write the chain rule for the composition of vector valued vector functions F and G. Pay attention to the order of the Jacobian matrices you write.

10. Let $F(x, y, z) = [3xy \quad y - z]^T$ and $G(x, y) = [x - 5y \quad xy \quad x - y]^T$ write the composition $(f \circ G)(x, y)$. Compute the Jacobian matrix of the composition in two ways: directly and by using the chain rule. Make sure the result you compute in both ways is identical.

11. Explain the need to order nodes when traversing the computation graph in the forward and backward passes.

12. Scalar computation graph:

    (a) Draw the computation graph for a regression network with three hidden units and one output unit. Assume logistic activation at hidden layer nodes and linear activation at output. Assume L2 loss.

    (b) Assume the current weight parameters guess is given by:

$$\begin{aligned} w_1 &= (0.01, 0.02, 0.03) \\ w_2 &= (0.03, 0.01, 0.02) \\ w_3 &= (0.02, 0.03, 0.01) \\ v &= (0.01, 0.02, 0.03, 0.04) \end{aligned}$$

    where w_1, w_2, w_3 are hidden unit weight vectors and v is the output unit weight vector. Assume the training data in the batch is given by:

$$\begin{aligned} x_1, y_1 &= [(1, 2), 8] \\ x_2, y_2 &= [(1, 3), 11] \\ x_3, y_3 &= [(2, 2), 10] \end{aligned}$$

    Compute the values at the network for the forward pass for each data point.

    (c) Compute the gradients of the loss function with respect to w_1, w_2, w_3, v due to all training examples using backpropagation.

    (d) Compute the gradients using the formula given in class and compare to your previous result.

13. Vector computation graph:

    (a) Let $f(x, y) = (2x + 3y)^2$. Compute $\nabla f(x, y)$.

    (b) Let $F(x, y) = \begin{bmatrix} x^2 + 2y \\ 3x + 4y^2 \end{bmatrix}$. Compute the Jacobian matrix $DF(1, 2)$.

    (c) Let $G(x) = \begin{bmatrix} x \\ x^2 \end{bmatrix}$ and $F(x, y)$ as in the previous question. Compute the Jacobian matrix $D(F \circ G)(2)$ both with and without the chain rule.

    (d) Repeat the scalar computation graph question (question 1) but this time using a vector notation. That is, draw the computation graph using vectors for input data and weights, perform a forward pass using vectors, and perform backpropagation by creating and multiplying Jacobian matrices instead of gradient vectors.

14. Assume a two layer classification network with a single output where the hidden units and output unit use a Sigmoid activation. Assume a training set $\{x^{(i)}, y^{(i)}\}_{i=1}^{m}$ where $x^{(i)} \in \mathbb{R}^n$ and $y^{(i)} \in \{0, 1\}$. Assuming L2 loss ($\frac{1}{2}\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2$) instead of binary cross entropy) derive the backpropagation update equations for the output weights in a similar way to what was done in class for the binary cross entropy loss case.

Part 2 (implementation questions)

Implement and test a three layer neural network (i.e. 2 hidden layers and one output layer) for a three class-classification using python and without using a GPU framework. Use categorical cross entropy for loss, sigmoid activation functions for hidden layers, and softmax for the output layer. The computation graph and its forward and backward traversal should be hard coded in the program without having to support dynamic configurations. Evaluate the network you implemented using similar steps to that of the previous assignment. Use two or more data sets (e.g. from the UCI Machine Learning Repository http://archive.ics.uci.edu/ml/). It is essential that you evaluate performance properly. Make sure to explain the results you obtain and do not unnecessarily repeat similar results. The code you write should be modular and well documented. The program needs to be written in Python. Your implementation should include the following components:

1. A loss function and an evaluation function.
2. A class for each node type with methods for computing a forward and backward pass.
3. Forward and backward traversal of the computation graph where data batches are pushed forward and gradient loss values are pushed backward.
4. Implementation of stochastic gradient descent with learning rate and decay parameters for updating model weights.

Submission instructions

Follow the submission instructions of assignment 1.