Department of Computer Science
Cochin University of Science and Technology
Deep Learning for Computer Vision - Assignment 01

Optimization and regularisation techniques activation and loss functions: Their Significance in Multilayer Perceptron

Aneetta Sara Shany
M.Tech Computer Science & Engineering - Software Engineering

April 3, 2023

Introduction

What is Multilayer perceptron ?

The multilayer perceptron (MLP) is the fundamental example of a deep neural network. In this assignment, we will be focusing more on different optimization techniques, loss functions, regularisation techniques, and activation functions and their significance in MLP.

MLP, a fully connected dense layers, which transform any input dimension to the desired dimension. It is a neural network that has multiple layers. To create a neural network we combine neurons together so that the outputs of some neurons are inputs of other neurons.

The MLP learning procedure is as follows:

- Starting with the input layer, propagate data forward to the output layer. This step is the forward propagation.
- Based on the output, calculate the error (the difference between the predicted and known outcome). The error needs to be minimized.
- Backpropagate the error. Find its derivative with respect to each weight in the network, and update the model.

Repeat the three steps given above over multiple epochs to learn ideal weights. Finally, the output is taken via a threshold function to obtain the predicted class labels.

Optimization techniques

Optimization algorithms or strategies are responsible for reducing the losses and to provide the most accurate results possible. The right optimization algorithm can reduce training time exponentially. They can also help to prevent overfitting and improve the stability of the training process.

- **Gradient Descent**

  Gradient Descent is used heavily in linear regression and classification algorithms. Back propagation in neural networks also uses a gradient descent algorithm.

- **Stochastic Gradient Descent**

  It's a variant of Gradient Descent which tries to update the model's parameters more frequently. In this, the model parameters are altered after computation of loss on each training example.

- **Adam**

  Adam is the best optimizers. If one wants to train the neural network in less time and more efficiently than Adam is the optimizer. It is a adaptive learning rate algorithms adjust the learning rate during training based on the gradients.

Regularisation techniques

Regularization techniques are used to prevent overfitting. Common techniques include L1 regularization (Lasso), L2 regularization (Ridge), and Dropout. Regularizing a model means changing its learning behavior during the training phase.

- **L1 regularization**
  L1 regularization, also known as "Lasso", adds a penalty on the sum of the absolute values of the model weights. This means that weights that do not contribute much to the model will be zeroed, which can lead to automatic feature selection
- **L2 regularization**
  L2 regularization, also called Ridge regularization, adds the square of the weights to the regularization term. This means that larger weights are reduced but not zeroed, which leads to models with fewer variables than L1 regularization but with more distributed weights.
- **Dropout**
  Adam is the best optimizers. If one wants to train the neural network in less time and more efficiently than Adam is the optimizer. It is a adaptive learning rate algorithms adjust the learning rate during training based on the gradients.

Activation functions

In Deep Learning, neurons receive input signals from the preceding layer of a neural network. A weighted sum of these signals is fed into the neuron's activation function, then the activation function's output is passed onto the next layer of the network. The function determines whether the neuron should be activated or not, based on whether each neuron's input is relevant for the model's prediction. Activation functions also help normalize the output of each neuron to a range between 1 and 0 or between -1 and 1.

* **Threshold functions**
  Threshold functions compute a different output signal depending on weather or not its input lies above or below a certain threshold. A binary step function is a threshold-based activation function. The problem with a step function is that it does not allow multi-value outputs—for example, it cannot support classifying the inputs into one of several categories.
* **Sigmoid functions**
  The sigmoid function is well-known among the data science community because of its use in logistic regression, one of the core machine learning techniques used to solve classification problems. The sigmoid function can accept any value, but always computes a value between 0 and 1.
* **ReLUs**
  The rectifier function does not have the same smoothness property as the sigmoid function. The neurons will only be deactivated if the output of the linear transformation is less than 0.

Loss functions

Loss functions are one of the most important aspects of neural networks, as they are directly responsible for fitting the model to the given training data. A loss function is a function that compares the target and predicted output values; measures how well the neural network models the training data. When training, we aim to minimize this loss between the predicted and target outputs. In supervised learning, there are two main types of loss functions — these correlate to the 2 major types of neural networks: regression and classification loss functions

* **Mean Squared Error**
  One of the most popular loss functions, MSE finds the average of the squared differences between the target and the predicted outputs. It is a regression loss function used in regression neural networks; given an input value, the model predicts a corresponding output value (rather than pre-selected labels). However, one disadvantage of this loss function is that it is very sensitive to outliers; if a predicted value is significantly greater than or less than its target value, this will significantly increase the loss.
* **Binary Cross-Entropy**
  This classification loss function is used in binary classification models; given an input, the neural network produces a vector of probabilities of the input belonging to various pre-set categories — can then select the category with the highest probability of belonging. It gives the probability value between 0 and 1 for a classification task. Cross-Entropy calculates the average difference between the predicted and actual probabilities.

. . . .