Exercises #7 - Ensemble Learning

Thales Menezes de Oliveira

Brazilian Center for Physics Research (CBPF)

April 1, 2022

1. It is okay to initialize all the weights to the same values as long as that value is selected randomly using He initialization?

A: The initialization of the weights to the same value, ths symmetry is not broken, in a way that we can say that all neurons in a given layer are equivalent. In this sense all weights should be sampled independently, in order to break these symmetries. For the weights initialization o the same values, all neurons in a given layer will always have the same weights, which it is like having just one neuron per layer. This prejudice to the training process and the fact that the backpropagation will not be able to break the symmetries, it is virtually impossible for such a configuration to converge to a good solution.

2. It is okay to initialize the bias terms to 0?

A: It is possible to initialize the bias terms to zero, or can it be initialized as the same way as the weights. It is expected not much difference between the two approaches.

3. Name three advantages of the ELU activation function over ReLU.

A: Since the ELU stands for exponential LU, it means that this function is smooth everywhere helping to speed up the gradient descent, since the abrupt changes in the z = 0 will make the GD bounce around it. Since it is always a smooth function, the derivative is not zero for all points, avoiding the dying units problems, which is one of the main drawbacks of the ReLU. At last, the ELU can take negative values, so the average output of the neurons in any given layer is typically closer to 0 than when using the ReLU activation function, which is not able to output negative values. This property of the ELU alleviate the vanishing gradient problem.

4. In which cases would you want to use each of the following activation functions: ELU, LeakyReLU (and its variants), ReLU, tanh, logistic , and softmax?

A: **LeakyReLU:** It can be used to get a fast neural network;
ReLU: The simplicity of this activation function makes it the preferred option for the majority of the NNs, despite they are generally outperformed by the ELU and leaky ReLU. The capability of outputting precisely zero can be useful for the autoencoders;
tanh: It is useful when the value of the output layer is supposed to be contained in the interval -1 to 1;
sigmoid: It can be used in the output layer to predict the probability of a instance to belong some specific class, specially for binary classifications. The VariationalAutoEncoders can use the sigmoid activation function in its hidden layers;
softmax: It can be used in a similar fashion of the sigmoid, but for the multiclassification, since it is capable to predict probabilities for mutually exclusive classes.

5. What may happen if you set *momentum* hyperparameter too close to 1, (e.g., 0.999999) when using a *MomentumOptimizer*

A: We can recall that the GD simply update the weights θ by directly subtracting the gradient of the cost function $J(\theta)$ with regards to the weights mutiplied by the learning rate η. The GD doesn't take in consideration what the earlier gradients were.

On the other hand, the consideration of the earlier gradients is a big deal to the momentum optimization. At each iterattion, it subtracts the local gradient from the *momentum* vector m, multiplied by the learning

rate η, and it updates the weights by simply adding this momentum vector. The momentum can escape from plateaus much faster than GD and it can reach the bottom of the valley faster.

Higher values for the momentum hyperparameter can cause it to shoot right past the minimum due its to momentum. Then it will slow down and come back, accelerate again, overshoot again, and so on. This oscillatory behaviour will happen many times before converging to the optimal value, so it can take a much longer time to converge, in comparison with a lower momentum value.

6. Name three ways you can produce a sparse model

A:

7. Does dropout slow down training? Does it slow down inference (i.e., making predictions on new instances)?

A: The dropout regularization does slow down the training, but it has no impact on the predictions since it is only used during the training.