Project : A GPU friendly framework for Multi Layer Perceptron*

Djoser SIMEU

October 2024

In the context of the M2 MOSIG lecture, "Mathematical Foundation of Machine Learning", we studied the Multi-Layer Perceptron (MLP) based on the studies of Rosenblatt [Ros58]. MLP is an acyclic fully connected neural network architecture where the neurons are structured in successive layers, beginning by an input layer and finishing with an output layer. As a machine learning model, the learnable parameters are represented by the weights of the connections between the layers of the MLP. The optimal weights of the model are found by solving the following optimization problem :

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \mathbf{L}_x(\mathbf{f}_{\mathbf{W}}(x), \mathbf{y}) \tag{1}$$

Where :

- $\mathbf{W}^*$ represent the optimal weights of the MLP model.
- x represent the set of data points on which we train our model.
- y represent the target of the data points of our training dataset.
- $\mathbf{f}_{\mathbf{W}}(x)$ represent the output layers of the MLP model with x as input layer and $\mathbf{W}$ as weights of the model.
- $\mathbf{L}_x$ represent the loss function used to compute the error of our model, in our case, this study is focus on classification problems so, we will use the cross-entropy loss function to measure the error of our model.

A way to solve this optimization problem is to use the gradient descent algorithm combined with the backpropagation principle [Ama93]. A classical way to manipulate our data during the training process as well as during the inference process is to represent them as matrices. In that context, for the GP-GPU programming project, I want to implement an optimized GPU computing framework for the deployment of the MLP architecture. There is a lot of challenge in the development of this project, I must optimize the GPU memory management during the training process to minimize time needed to access the learnable parameters of the models. In terms of computation, with the large number of parameters of an MLP model, we could gain in terms of performance by using a parallelize computation provided by the GPU computing implementation compare to a CPU computing implementation [OJ04].

*This work is partially supported by the French National Research Agency in the framework of the "France 2030" program (ANR-11-LABX-0025-01) for the LabEx PERSYVAL.

References

[Ama93] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196, 1993.

[OJ04] Kyoung-Su Oh and Keechul Jung. Gpu implementation of neural networks. *Pattern Recognition*, 37(6):1311–1314, 2004.

[Ros58] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.