**PROBABILISTIC ARTIFICIAL INTELLIGENCE - TASK 2 (BAYESIAN NEURAL NETWORK)**

The first step was about implementing the layers' structure of the network. First of all, I needed to initialize the parameters: both weight\_mu and bias\_mu were set to be uniformly distributed in the range (-0.0005, +0.0005), while weight\_logsigma and bias\_logsigma were set to have a constant value of -2.55 for each element (all these hyperparameters were tuned by trial and error). The prior, instead, was set to be a Gaussian distribution with mean=0 and std=0.1.

In the forward() method of the BayesianLayer class, I performed the reparametrization trick, which allowed me to sample the output weight and bias of each layer from a simpler distribution (with mean=0 and std=1). In this method, I computed also the KL loss of each layer by taking the difference between the log prior and the log posterior of both the weight and the bias, and then summing them together.

Predictions of the class probabilities were made by considering a Softmax layer at the very end of the network. The loss function was computed by adding the cross-entropy loss with the KL loss. Moreover, I implemented also an additional function which computes the overall ELBO and allows you to sample n times from each of its components (and then averages the results).

The best results were obtained by training the network on 35 epochs, with a learning rate of 5e-4 and a batch size of 128 (and sampling 6 times from the ELBO). With these parameters I was able to score 2.1208 on the public leaderboard.