Master 1 SD, SIME
Master 2 SIME

Apprentissage profond

Nom : COELHO FERNANDES
Prénom : Rodrigo César

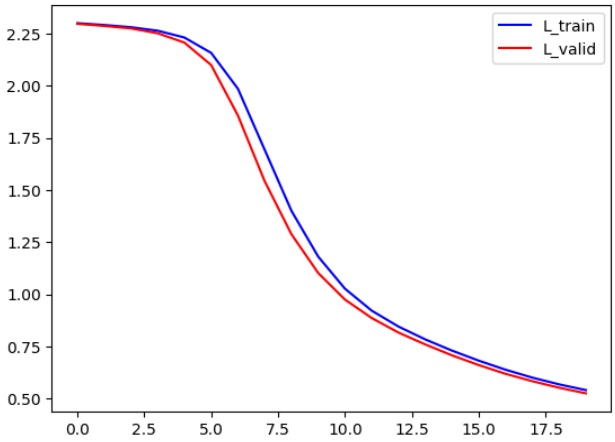
Séance de TP 2
Perceptron multicouches et descente de gradient

1- Mettre en évidence l'influence des paramètres de l'algorithme SGD (batch_size, learning_rate, epochs)

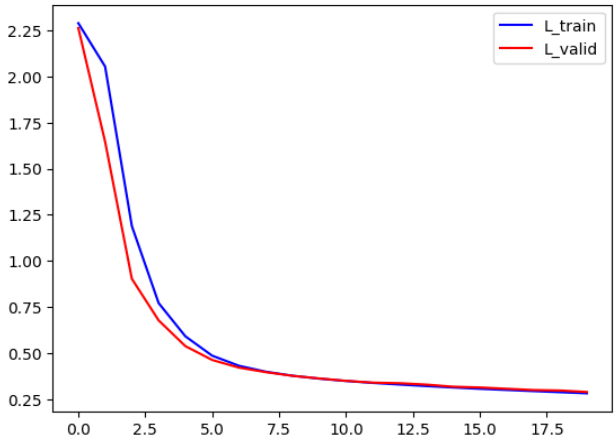
Batch Size:

Batches that are too small introduce noise into convergence. We observe a more turbulent and unstable start in the example with batch_size = 16.

batch_size = 64



batch_size = 16

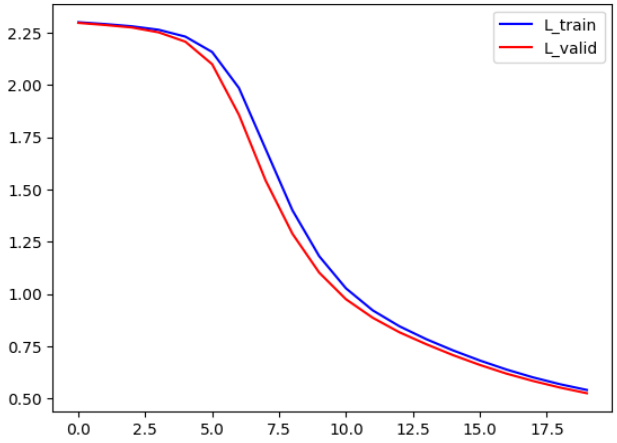


However, in deep learning, instability can sometimes be beneficial for convergence, as it helps avoid getting stuck in overly local minima. This seems to be the case in this example, where the model reached better Loss values during training and also performed better after testing (85.34% for a bigger batch vs 92.03% for the smaller one).

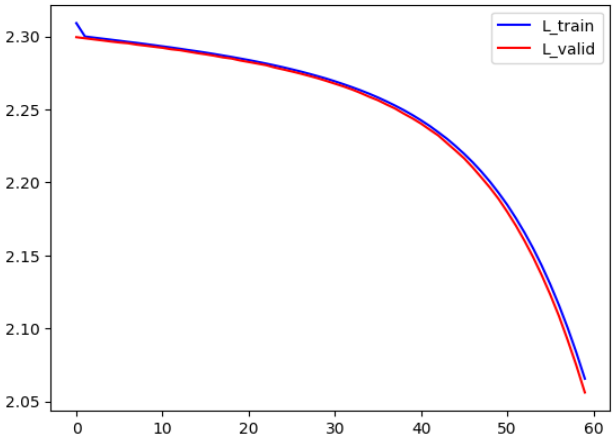
Learning Rate:

A low λ allows convergence but at a very slow pace. However, in this example, the learning rate was too low to reach a convergence point before all epochs were completed.

λ = 1e-2



λ = 1e-3

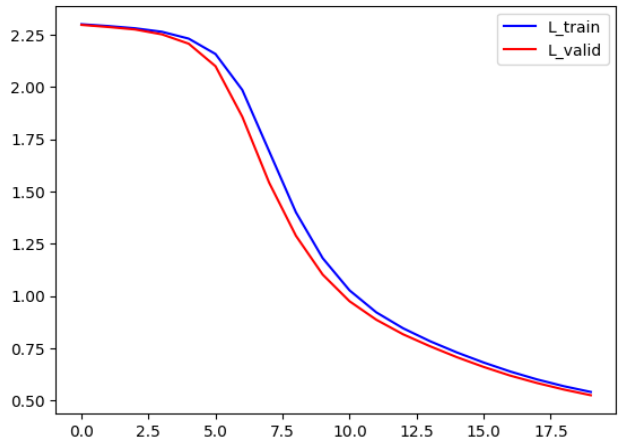


On the other hand, a high λ prevents convergence. A possible solution to this problem could be automatic adaptation of the gradient step, where if there is no improvement for a certain number of epochs, λ is halved ($\lambda = \lambda/2$).

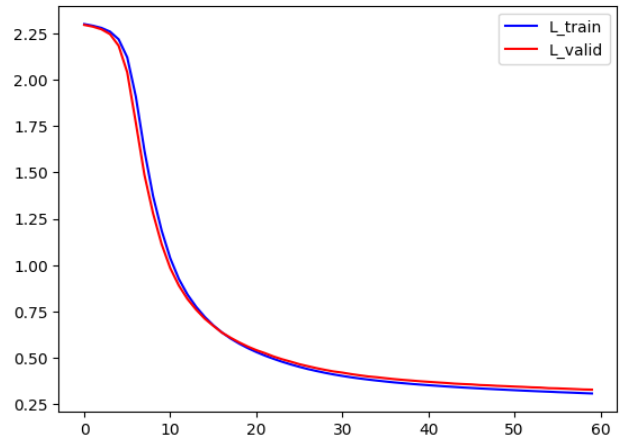
Epochs:

Increasing the number of epochs while keeping other parameters constant allows the model to continue reducing the Loss value. However, if the Loss reaches a plateau where it no longer decreases significantly as epochs progress, further training is no longer beneficial.

Epochs = 20



Epochs = 60

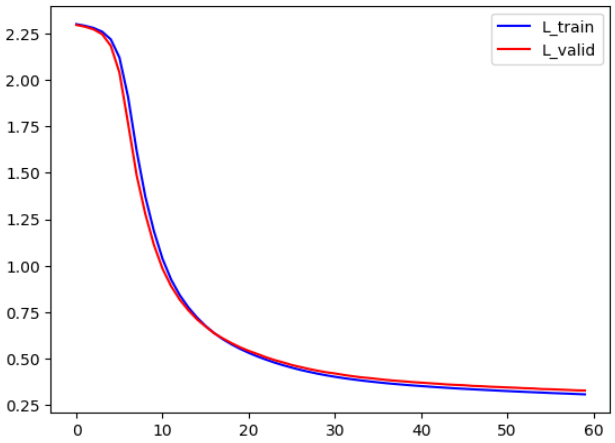


In this example, we can see that increasing the number of epochs to 60 allowed the model to achieve a lower Loss value during training (from 85,34% to 91.08%).

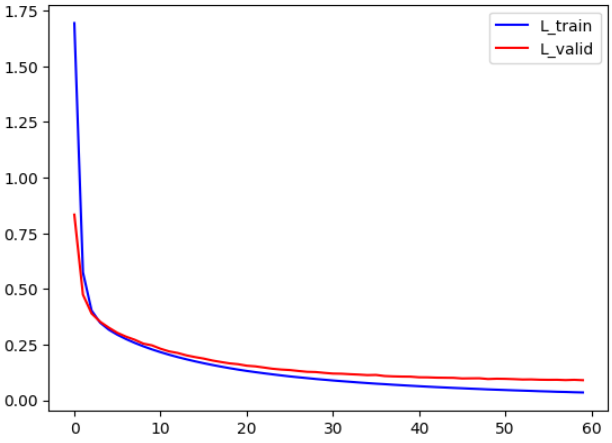
2- Mettre en évidence l'apport de la fonction d'activation RELU() par rapport à la fonction Sigmoïde() (vitesse de convergence, loss, accuracy)

The convergence speed of the RELU() function was significantly faster than that of the sigmoid() function, reaching the convergence point around epochs 3 and 10, respectively. From these points onward, the decrease in loss becomes considerably smaller as epochs progress.

Sigmoid()



RELU()

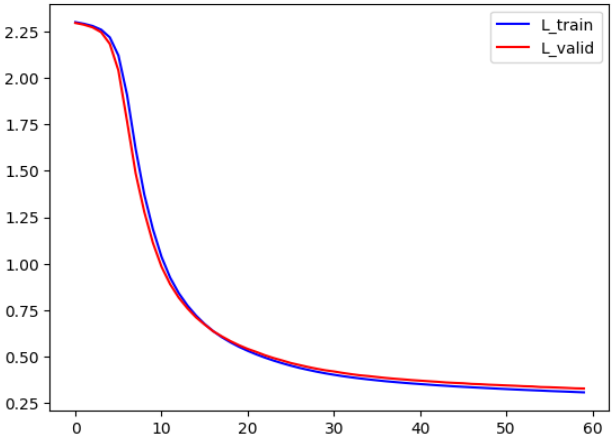


It is observed that the Loss reached lower values for the RELU() function. Even though it exhibited greater divergence between training and validation results, RELU() achieved better accuracy on the test data (95.83% vs. 91.08%), proving to be the most suitable activation function for this model.

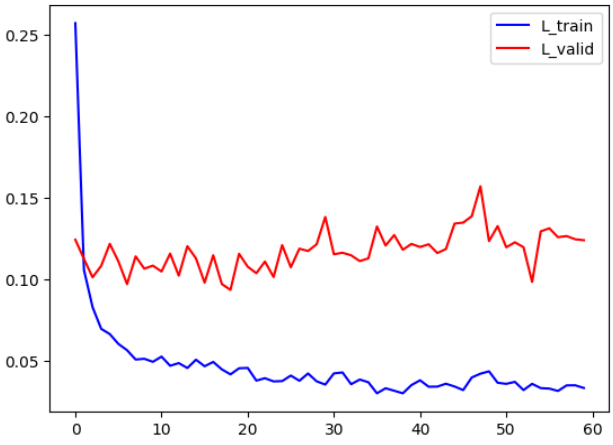
3- Comparer SGD et ADAM en termes de convergence (temps et Loss)

The ADAM method exhibited highly unstable convergence with significant variation between training and validation loss. However, it achieved a very high final accuracy (97,28% on the 19th epoch), reflecting the low loss values obtained. Therefore, using ADAM in this neural network is recommended for better results.

SGD - λ = 1e-2

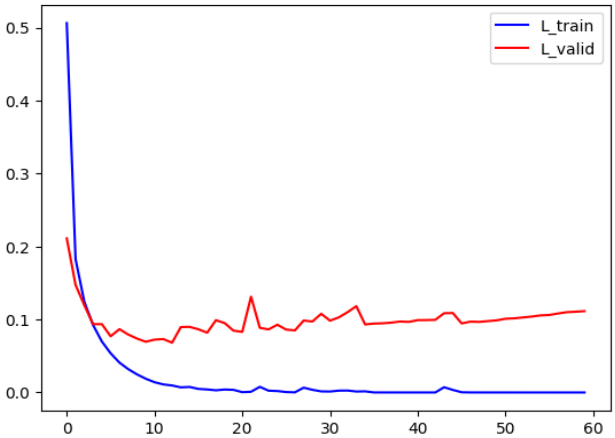


ADAM - λ = 1e-2



Reducing the learning rate ensured better stability across epochs, as expected, and improved final accuracy (98,00 on the 13th epoch) since convergence was reached quickly. However, the validation loss tended to increase after convergence was achieved, which may indicate overfitting. For this reason, it is advisable to use early stopping or retain the model with the lowest validation loss, as implemented in the algorithm.

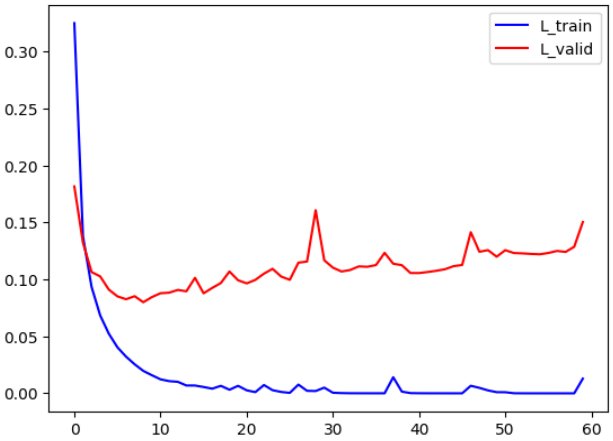
ADAM - λ = 1e-3



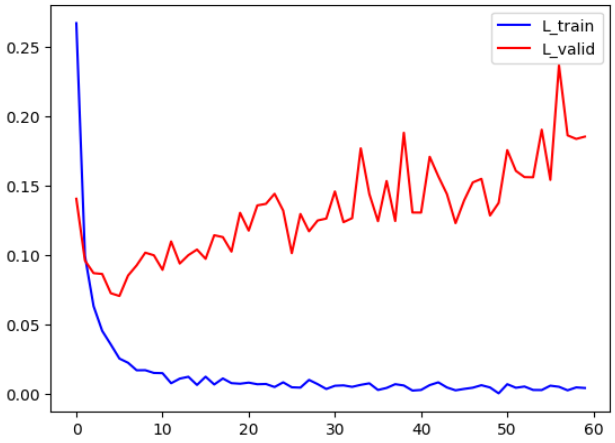
4- Mettre en évidence l'influence de l'architecture et du nombre de paramètres

The architectures with 3 and 4 layers achieved faster convergence, around the 6th epoch, compared to the 10th epoch for the 2-layer architecture. However, the accuracy for all three tests was quite similar, around 97.9%.

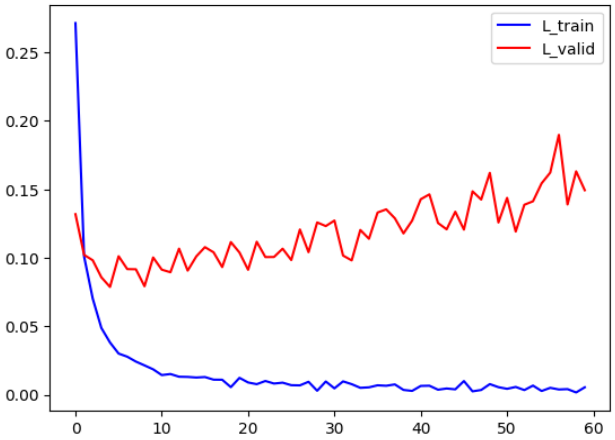
2 couches 256 – 10



3 couches 512 – 128 – 10



4 couches 512 – 256 – 128 – 10



It is also important to note the number of parameters of each of the neural network architectures. The 2, 3 and 4 layer networks presented, respectively, 171730, 468874 and 567434 parameters. The number of parameters has a direct relationship with the calculation time and computational cost.

Another noteworthy behavior is the stability of the loss evolution. Architectures with fewer layers exhibited more stable loss curves. Also, once again, we observe the strong effect of overfitting, causing the validation loss to increase with each epoch across all architectures.