# Project 1

The code is divided in two files. The first one “Project1.py” has the training loops for the different architectures. The second file “comparisonNets.py” contains the three architectures.

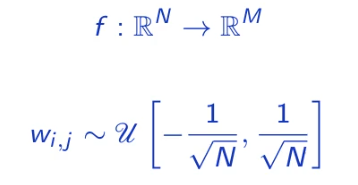
In order to launch the code you simple run the main script “Project1.py” and it outputs the result of the training and testing of the three architectures in the console.

The three architectures are composed as follow :

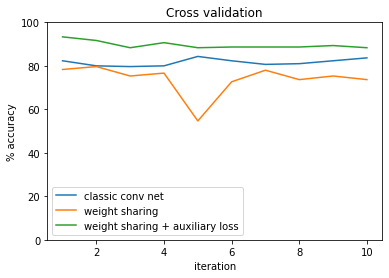
1. The input tensor is seen has a 2 channels image. There is a feature extraction with two convolutionals layers and then a fully connected head that predict if the first digit is lesser or equal to the second digit.

1. Each channel of the input tensor is processed by the same network (weight sharing) which try to identify the number with a softmax function with 10 output. Then a fully connected layer try to predict if the first digit is lesser or equal to the second digit.
2. Same as 2, but we use auxiliary loss to force the network to optimize the digit recognition part.

Before each training, we call a function “initParameter” that initialize the weights according to the following rule :



Finally, we get those results :



We see that the weight sharing and the auxiliary loss outperform the two other architectures. It might be because we ensure that the network learn to recognize a number and then compare the intermediate result to reach to a conclusion. In the two other cases we simply try to optimize a model in order to minimize the binary cross entropy loss of the final decision which seems to be capped around 80% with the given dataset.

A future work could be to tune the loss in order to give a bigger importance to the recognition of the digit first and then optimize the loss of the final decision.